[CHAPTER [1] 8](#_Toc438046734)

[HTML Introduction: 8](#_Toc438046735)

[Basic HTML Document: 8](#_Toc438046736)

[Web Browsers: 9](#_Toc438046737)

[HTML Tags: 9](#_Toc438046738)

[HTML Document  Structure: 9](#_Toc438046739)

[The <!DOCTYPE> Declaration: 10](#_Toc438046740)

[Common Declarations: 10](#_Toc438046741)

[CHAPTER [2] 11](#_Toc438046742)

[HTML Editors: 11](#_Toc438046743)

[Write HTML Using Notepad or TextEdit: 11](#_Toc438046744)

[Step 1: Open Notepad: 11](#_Toc438046745)

[Step 2: Write Some HTML: 11](#_Toc438046746)

[Step 3: Save the HTML Page: 12](#_Toc438046747)

[Step 4: View HTML Page in Your Browser: 13](#_Toc438046748)

[CHAPTER [3] 14](#_Toc438046749)

[HTML Documents: 14](#_Toc438046750)

[Heading Tags: 14](#_Toc438046751)

[Paragraph Tag: 15](#_Toc438046752)

[Line Break Tag: 15](#_Toc438046753)

[HTML Links: 16](#_Toc438046754)

[HTML Images: 16](#_Toc438046755)

[Centering Content: 17](#_Toc438046756)

[Horizontal Lines: 17](#_Toc438046757)

[Preserve Formatting: 18](#_Toc438046758)

[Nonbreaking Spaces: 19](#_Toc438046759)

[CHAPTER [4] 20](#_Toc438046760)

[HTML Elements: 20](#_Toc438046761)

[HTML Tag vs. Element: 20](#_Toc438046762)

[Nested HTML Elements: 21](#_Toc438046763)

[Don't Forget the End Tag: 21](#_Toc438046764)

[Empty HTML Elements: 22](#_Toc438046765)

[HTML Tip: Use Lowercase Tags: 22](#_Toc438046766)

[CHAPTER [5] 23](#_Toc438046767)

[HTML Attributes: 23](#_Toc438046768)

[Core Attributes: 24](#_Toc438046769)

[The id Attribute: 24](#_Toc438046770)

[The title Attribute: 24](#_Toc438046771)

[The class Attribute: 25](#_Toc438046772)

[The lang Attribute: 25](#_Toc438046773)

[The href Attribute: 26](#_Toc438046774)

[Size Attributes: 26](#_Toc438046775)

[The alt Attribute: 27](#_Toc438046776)

[We Suggest: Always Use Lowercase Attributes: 27](#_Toc438046777)

[Single or Double Quotes?: 27](#_Toc438046778)

[Chapter Summary: 28](#_Toc438046779)

[HTML Styling: 28](#_Toc438046780)

[The HTML Style Attribute: 28](#_Toc438046781)

[HTML Text Color: 28](#_Toc438046782)

[HTML Fonts: 29](#_Toc438046783)

[HTML Text Size: 29](#_Toc438046784)

[HTML Text Alignment: 29](#_Toc438046785)

[CHAPTER [6] 31](#_Toc438046786)

[HTML Formatting: 31](#_Toc438046787)

[HTML Formatting Elements: 31](#_Toc438046788)

[HTML Bold and Strong Formatting: 31](#_Toc438046789)

[Underlined Text: 32](#_Toc438046790)

[HTML Small Formatting: 32](#_Toc438046791)

[HTML Marked Formatting: 33](#_Toc438046792)

[HTML Deleted Formatting: 33](#_Toc438046793)

[HTML Inserted Formatting: 33](#_Toc438046794)

[HTML Subscript Formatting: 34](#_Toc438046795)

[HTML Superscript Formatting: 34](#_Toc438046796)

[Strike Text: 34](#_Toc438046797)

[Monospaced Font: 35](#_Toc438046798)

[Emphasized Text: 35](#_Toc438046799)

[CHAPTER [7] 36](#_Toc438046800)

[HTML Quotation: 36](#_Toc438046801)

[HTML <q> for Short Quotations: 36](#_Toc438046802)

[HTML <blockquote> for Long Quotations: 36](#_Toc438046803)

[HTML <abbr> for Abbreviations: 37](#_Toc438046804)

[Text Direction: 37](#_Toc438046805)

[Text Citations: 38](#_Toc438046806)

[HTML <address> for Contact Information: 38](#_Toc438046807)

[HTML Computer Code Formatting: 39](#_Toc438046808)

[HTML Code Formatting: 39](#_Toc438046809)

[Keyboard Text: 39](#_Toc438046810)

[HTML Sample Formatting: 40](#_Toc438046811)

[Programming Variables: 40](#_Toc438046812)

[HTML Variable Formatting: 41](#_Toc438046813)

[CHAPTER [8] 42](#_Toc438046814)

[HTML Comments: 42](#_Toc438046815)

[Conditional Comments: 42](#_Toc438046816)

[Using Comment Tag: 43](#_Toc438046817)

[Commenting Script Code: 43](#_Toc438046818)

[Commenting Style Sheets: 44](#_Toc438046819)

[CHAPTER [9] 45](#_Toc438046820)

[HTML Style Sheet: 45](#_Toc438046821)

[CSS Syntax: 45](#_Toc438046822)

[Styling HTML with CSS: 45](#_Toc438046823)

[External Style Sheet: 45](#_Toc438046824)

[Internal Style Sheet: 47](#_Toc438046825)

[Inline Style Sheet: 47](#_Toc438046826)

[CSS Fonts: 48](#_Toc438046827)

[The CSS Box Model: 49](#_Toc438046828)

[The id Attribute: 51](#_Toc438046829)

[The class Attribute: 51](#_Toc438046830)

[Deprecated Tags and Attributes in HTML5: 52](#_Toc438046831)

[CHAPTER [10] 53](#_Toc438046832)

[HTML Links: 53](#_Toc438046833)

[HTML Links – Hyperlinks: 53](#_Toc438046834)

[HTML Links – Syntax: 53](#_Toc438046835)

[Local Links: 53](#_Toc438046836)

[Example: 53](#_Toc438046837)

[HTML Links – Colors: 54](#_Toc438046838)

[HTML Links - The target Attribute: 55](#_Toc438046839)

[HTML Links - Image as Link: 56](#_Toc438046840)

[HTML Links - Create a Bookmark: 56](#_Toc438046841)

[CHAPTER [11] 59](#_Toc438046842)

[HTML Images: 59](#_Toc438046843)

[HTML Images Syntax: 59](#_Toc438046844)

[Insert Image: 59](#_Toc438046845)

[Set Image Location: 60](#_Toc438046846)

[Set Image Width/Height: 60](#_Toc438046847)

[Set Image Border: 61](#_Toc438046848)

[Set Image Alignment: 61](#_Toc438046849)

[Images on Another Server: 61](#_Toc438046850)

[Using an Image as a Link: 62](#_Toc438046851)

[Image Maps: 62](#_Toc438046852)

[Image Floating: 63](#_Toc438046853)

[CHAPTER [12] 65](#_Toc438046854)

[HTML Tables: 65](#_Toc438046855)

[Defining HTML Tables: 65](#_Toc438046856)

[Table Heading: 66](#_Toc438046857)

[Cellpadding and Cellspacing Attributes: 67](#_Toc438046858)

[Colspan and Rowspan Attributes: 68](#_Toc438046859)

[Tables Backgrounds: 69](#_Toc438046860)

[Table Height and Width: 70](#_Toc438046861)

[Table Caption: 71](#_Toc438046862)

[Table Header, Body, and Footer: 71](#_Toc438046863)

[Nested Tables: 73](#_Toc438046864)

[CHAPTER [13] 75](#_Toc438046865)

[HTML Lists: 75](#_Toc438046866)

[Unordered HTML Lists: 75](#_Toc438046867)

[Unordered HTML Lists - The Style Attribute: 76](#_Toc438046868)

[Disc: 76](#_Toc438046869)

[Circle: 76](#_Toc438046870)

[Square: 77](#_Toc438046871)

[None: 77](#_Toc438046872)

[Ordered HTML Lists: 78](#_Toc438046873)

[Ordered HTML Lists - The Type Attribute: 78](#_Toc438046874)

[Numbers: 79](#_Toc438046875)

[Uppercase Letters: 79](#_Toc438046876)

[Lowercase Letters: 79](#_Toc438046877)

[Uppercase Roman Numbers: 80](#_Toc438046878)

[Lowercase Roman Numbers: 80](#_Toc438046879)

[HTML Description Lists: 80](#_Toc438046880)

[Nested HTML Lists: 81](#_Toc438046881)

[Horizontal Lists: 82](#_Toc438046882)

[CHAPTER [14] 84](#_Toc438046883)

[HTML Blocks: 84](#_Toc438046884)

[Block Elements: 84](#_Toc438046885)

[Inline Elements: 84](#_Toc438046886)

[Grouping HTML Elements: 84](#_Toc438046887)

[The <div> tag: 84](#_Toc438046888)

[The <span> tag: 85](#_Toc438046889)

[CHAPTER [15] 87](#_Toc438046890)

[HTML Layouts: 87](#_Toc438046891)

[HTML Layout - Using Tables: 87](#_Toc438046892)

[Multiuple Columns Layout - Using Tables: 88](#_Toc438046893)

[HTML Layouts - Using DIV, SPAN: 89](#_Toc438046894)

[CHAPTER [16] 91](#_Toc438046895)

[HTML Iframes: 91](#_Toc438046896)

[Iframe Syntax: 91](#_Toc438046897)

[Iframe - Remove the Border: 91](#_Toc438046898)

[Use iframe as a Target for a Link: 92](#_Toc438046899)

[CHAPTER [17] 93](#_Toc438046900)

[HTML Javascript: 93](#_Toc438046901)

[External Javascript: 93](#_Toc438046902)

[Internal Script: 94](#_Toc438046903)

[Event Handlers: 94](#_Toc438046904)

[Hide Scripts from Older Browsers: 95](#_Toc438046905)

[The <noscript> Element: 95](#_Toc438046906)

[Default Scripting Language: 96](#_Toc438046907)

[JavaScript can change HTML content: 96](#_Toc438046908)

[JavaScript can change HTML styles: 97](#_Toc438046909)

[JavaScript can change HTML attributes: 97](#_Toc438046910)

[CHAPTER [17] 99](#_Toc438046911)

[HTML Head: 99](#_Toc438046912)

[The HTML <head> Element: 99](#_Toc438046913)

[Omitting <html> and <body>? 99](#_Toc438046914)

[Omitting <head>: 99](#_Toc438046915)

[The HTML <title> Element: 100](#_Toc438046916)

[The HTML <style> Element: 100](#_Toc438046917)

[The HTML <link> Element: 101](#_Toc438046918)

[The HTML <meta> Element: 101](#_Toc438046919)

[CHAPTER [18] 103](#_Toc438046920)

[HTML Forms: 103](#_Toc438046921)

[The <input> Element 103](#_Toc438046922)

[Text Input 103](#_Toc438046923)

[Radio Button Input 103](#_Toc438046924)

[The Submit Button 104](#_Toc438046925)

[The Action Attribute 104](#_Toc438046926)

[The Method Attribute 104](#_Toc438046927)

[When to Use GET? 105](#_Toc438046928)

[When to Use POST? 105](#_Toc438046929)

[The Name Attribute 105](#_Toc438046930)

[Grouping Form Data with <fieldset> 105](#_Toc438046931)

[HTML Form Attributes 106](#_Toc438046932)

[HTML Form Elements: 106](#_Toc438046933)

[The <input> Element 106](#_Toc438046934)

[The <select> Element (Drop-Down List) 106](#_Toc438046935)

[The <textarea> Element 107](#_Toc438046936)

[The <button> Element 107](#_Toc438046937)

[HTML5 Form Elements 107](#_Toc438046938)

[HTML5 <datalist> Element 107](#_Toc438046939)

[HTML5 <keygen> Element 108](#_Toc438046940)

[HTML5 <output> Element 108](#_Toc438046941)

[Input Type: text 109](#_Toc438046942)

[Input Type: password 109](#_Toc438046943)

[Input Type: submit 109](#_Toc438046944)

[Input Type: radio 110](#_Toc438046945)

[Input Type: checkbox 110](#_Toc438046946)

[Input Type: button 110](#_Toc438046947)

[HTML5 Input Types 111](#_Toc438046948)

[Input Type: number 111](#_Toc438046949)

[Input Type: date 111](#_Toc438046950)

[Input Type: color 112](#_Toc438046951)

[Input Type: range 112](#_Toc438046952)

[Input Type: month 112](#_Toc438046953)

[Input Type: week 113](#_Toc438046954)

[Input Type: time 113](#_Toc438046955)

[Input Type: datetime 113](#_Toc438046956)

[Input Type: datetime-local 113](#_Toc438046957)

[Input Type: email 114](#_Toc438046958)

[Input Type: search 114](#_Toc438046959)

[Input Type: tel 114](#_Toc438046960)

[Input Type: url 115](#_Toc438046961)

[HTML Input Attributes: 115](#_Toc438046962)

[The value Attribute 115](#_Toc438046963)

[The readonly Attribute 115](#_Toc438046964)

[The disabled Attribute 116](#_Toc438046965)

[The size Attribute 116](#_Toc438046966)

[The maxlength Attribute 116](#_Toc438046967)

[The autocomplete Attribute 117](#_Toc438046968)

[The novalidate Attribute 117](#_Toc438046969)

[The autofocus Attribute 118](#_Toc438046970)

[The form Attribute 118](#_Toc438046971)

[The formaction Attribute 118](#_Toc438046972)

[The formenctype Attribute 119](#_Toc438046973)

[The formmethod Attribute 119](#_Toc438046974)

[The formnovalidate Attribute 120](#_Toc438046975)

[The formtarget Attribute 120](#_Toc438046976)

[The height and width Attributes 120](#_Toc438046977)

[The list Attribute 121](#_Toc438046978)

[The min and max Attributes 121](#_Toc438046979)

[The multiple Attribute 121](#_Toc438046980)

[The pattern Attribute 122](#_Toc438046981)

[The placeholder Attribute 122](#_Toc438046982)

[The required Attribute 122](#_Toc438046983)

[The step Attribute 123](#_Toc438046984)

**HTML Tutorial**

HTML stands for **H**yper **T**ext **M**arkup **L**anguage, which is the most widely used language on Web to develop web pages.

HTML is a **markup** language for **describing** web documents (web pages).

* A markup language is a set of **markup tags**
* HTML documents are described by **HTML tags**
* Each HTML tag **describes** different document content

HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

# Audience

This tutorial is designed for the aspiring Web Designers and Developers with a need to understand the HTML in enough detail along with its simple overview, and practical examples. This tutorial will give you enough ingredients to start with HTML from where you can take yourself at higher level of expertise.

# Prerequisites

Before proceeding with this tutorial you should have a basic working knowledge with Windows or Linux operating system, additionally you must be familiar with:

* Experience with any text editor like notepad, notepad++, or Editplus etc.
* How to create directories and files on your computer.
* How to navigate through different directories.
* How to type content in a file and save them on a computer.
* Understanding about images in different formats like JPEG, PNG format.

CHAPTER [1]

# HTML Introduction:

# HTML stands for Hypertext Markup Language, and it is the most widely used language to write Web Pages.

* **Hypertext** refers to the way in which Web pages (HTML documents) are linked together. Thus the link available on a webpage are called Hypertext.
* As its name suggests, HTML is a **Markup Language** which means you use HTML to simply "mark up" a text document with tags that tell a Web browser how to structure it to display.

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers.

Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

# Basic HTML Document:

In its simplest form, following is an example of an HTML document:

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <title>This is document title</title>  </head>  <body>  <h1>This is a heading</h1>  <p>Document content goes here.....</p>  </body>  </html> |

## Example Explained:

* The **DOCTYPE** declaration defines the document type to be HTML
* The text between **<html>** and **</html>** describes an HTML document
* The text between **<head>** and **</head>** provides information about the document
* The text between **<title>** and **</title>** provides a title for the document
* The text between **<body>** and **</body>** describes the visible page content
* The text between **<h1>** and **</h1>** describes a heading
* The text between **<p>** and **</p>** describes a paragraph

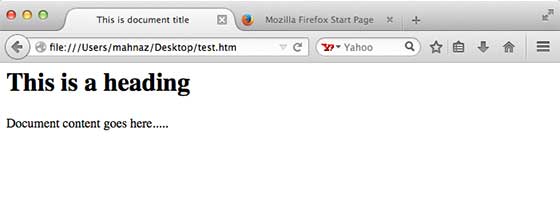
Using this description, a web browser can display a document with a heading and a paragraph.

Either you can use **Notepad++**  to check the result of this HTML code, or let's save it in an HTML file **test.htm**. Finally open it using a web browser like Internet Explorer or Google Chrome, or Firefox etc. It must show the following output:

# Web Browsers:

The purpose of a web browser (Chrome, IE, Firefox, Safari) is to read HTML documents and display them.

The browser does not display the HTML tags, but uses them to determine how to display the document:



# HTML Tags:

As told earlier, HTML is a markup language and makes use of various tags to format the content. These tags are enclosed within angle braces **<Tag Name>**. Except few tags, most of the tags have their corresponding closing tags. For example **<html>** has its closing tag **</html>** and **<body>** tag has its closing tag **</body>** tag etc

# HTML Document  Structure:

<html>

<head>

Document header related tags

</head>

<body>

Document body related tags

</body>

</html>

We will study all the header and body tags in subsequent chapters, for now let's see what is document declaration tag.

Only the <body> area is displayed by the browser.

# The <!DOCTYPE> Declaration:

The <!DOCTYPE> declaration tag is used by the web browser to understand the version of the HTML used in the document.

The <!DOCTYPE> declaration helps the browser to display a web page correctly.

There are different document types on the web.

To display a document correctly, the browser must know both type and version.

The doctype declaration is not case sensitive. All cases are acceptable:

1. <!DOCTYPE html>
2. <!DOCTYPE HTML>
3. <!doctype html>
4. <!Doctype Html>

# Common Declarations:

### HTML5:

<!DOCTYPE html>

### HTML 4.01:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.Google.org/TR/html4/loose.dtd">

### XHTML 1.0:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w Google 3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

CHAPTER [2]

# HTML Editors:

# Write HTML Using Notepad or TextEdit:

HTML can be edited by using professional HTML editors like:

* Microsoft WebMatrix
* Sublime Text

However, for learning HTML we recommend a text editor like Notepad (PC) or TextEdit (Mac).

We believe using a simple text editor is a good way to learn HTML.

Follow the 4 steps below to create your first web page with Notepad.

# Step 1: Open Notepad:

To open Notepad in Windows 7 or earlier:

Click **Start** (bottom left on your screen). Click **All Programs**. Click **Accessories**. Click **Notepad**.

To open Notepad in Windows 8 or later:

Open the **Start Screen** (the window symbol at the bottom left on your screen). Type **Notepad**.

# Step 2: Write Some HTML:

Write or copy some HTML into Notepad.

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My First Heading</h1>  
  
<p>My first paragraph.</p>  
  
</body>

</html>

**

# Step 3: Save the HTML Page:

Save the file on your computer.

Select **File > Save as** in the Notepad menu.

Name the file "index.html" or any other name ending with html or htm.

UTF-8 is the preferred encoding for HTML files.

ANSI encoding covers US and Western European characters only.



You can use either .htm or .html as file extension. There is no difference, it is up to you.

# Step 4: View HTML Page in Your Browser:

***Note:*** *Open the saved HTML file in your favorite browser. The result will look much like this :*



***Note:*** *To open a file in a browser, double click on the file, or right-click, and choose open with.*

CHAPTER [3]

# HTML Documents:

All HTML documents must start with a type declaration: **<!DOCTYPE html>**.

The HTML document itself begins with **<html>** and ends with **</html>**.

The visible part of the HTML document is between **<body>** and **</body>**.

### **Example:**

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My First Heading</h1>  
  
<p>My first paragraph.</p>  
  
</body>  
</html>

# Heading Tags:

HTML Heading are defined with the **<h>** tag:

Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements **<h1>, <h2>, <h3>, <h4>, <h5>, and <h6>**. While displaying any heading, browser adds one line before and one line after that heading.

HTML headings are defined with the **<h1>** to **<h6>** tags:

### **Example:**

<!DOCTYPE html>  
<html>  
<body>

<h1>This is a heading1</h1>  
<h2>This is a heading2</h2>  
<h3>This is a heading3</h3>  
<h4>This is a heading4</h4>  
<h5>This is a heading5</h5>  
<h6>This is a heading6</h6>

</body>  
</html>

# Paragraph Tag:

The **<p>** tag offers a way to structure your text into different paragraphs. Each paragraph of text should go in between an opening <p> and a closing </p> tag as shown below in the example:

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Paragraph Example</title>

</head>

<body>

<p>Here is a first paragraph of text.</p>

<p>Here is a second paragraph of text.</p>

<p>Here is a third paragraph of text.</p>

</body>

</html>

# Line Break Tag:

HTML Line Break Tag are defined with the **<br />** tag:

Whenever you use the **<br />** element, anything following it starts from the next line. This tag is an example of an **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The <br /> tag has a space between the characters **br** and the forward slash. If you omit this space, older browsers will have trouble rendering the line break, while if you miss the forward slash character and just use <br> it is not valid in XHTML.

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Line Break Example</title>

</head>

<body>

<p>Hello<br />

You delivered your assignment ontime.<br />

Thanks<br />

Mahnaz</p>

</body>

</html>

# HTML Links:

HTML links are defined with the **<a>** tag:

### **Example:**

<!DOCTYPE html>

<html>

<body>

<a href="http://www.schoolconnect.me">This is a link</a>

</body>

</html>

# HTML Images:

HTML images are defined with the **<img>** tag.

The source file (**src**), alternative text (**alt**), and size (**width** and **height**) are provided as **attributes**:

### **Example:**

<!DOCTYPE html>

<html>

<body>

<img src="schoolconnect.jpg" alt="www.schoolconnect.me" width="104" height="142">

</body>

</html>

# Centering Content:

You can use **<center>** tag to put any content in the center of the page or any table cell.

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Centring Content Example</title>

</head>

<body>

<p>This text is not in the center.</p>

<center>

<p>This text is in the center.</p>

</center>

</body>

</html>

# Horizontal Lines:

Horizontal lines are used to visually break up sections of a document. The **<hr/>**tag creates a line from the current position in the document to the right margin and breaks the line accordingly.

For example you may want to give a line between two paragraphs as in the given example below:

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Horizontal Line Example</title>

</head>

<body>

<p>This is paragraph one and should be on top</p>

<hr />

<p>This is paragraph two and should be at bottom</p>

</body>

</html>

Again **<hr />** tag is an example of the **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The <hr /> element has a space between the characters **hr** and the forward slash. If you omit this space, older browsers will have trouble rendering the horizontak line, while if you miss the forward slash character and just use <hr> it is not valid in XHTML.

# Preserve Formatting:

Sometimes you want your text to follow the exact format of how it is written in the HTML document. In those cases, you can use the preformatted tag <pre>.

Any text between the opening <pre> tag and the closing </pre> tag will preserve the formatting of the source document.

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Preserve Formatting Example</title>

</head>

<body>

<pre>

function testFunction( strText ){

alert (strText)

}

</pre>

</body>

</html>

***Note:*** *Try using same code without keeping it inside <pre>...</pre> tags.*

# Nonbreaking Spaces:

Suppose you want to use the phrase "12 Angry Men." Here you would not want a browser to split the "12, Angry" and "Men" across two lines:

In cases where you do not want the client browser to break text, you should use a nonbreaking space entity **&nbsp;** instead of a normal space. For example, when coding the "12 Angry Men" in a paragraph, you should use something similar to the following code:

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Nonbreaking Spaces Example</title>

</head>

<body>

<p>An example of this technique appears in the movie "12&nbsp;Angry&nbsp;Men."</p>

</body>

</html>

CHAPTER [4]

# HTML Elements:

HTML **documents** are made up by HTML **elements**.

# An HTML element is defined by a starting tag. If the element contains other content, it ends with a closing tag, where the element name is preceded by a forward slash as shown below with few tags:

HTML elements are written with a **start** tag, with an **end** tag, with the **content** in between:

<tagname>content</tagname>

The HTML **element** is everything from the start tag to the end tag:

|  |  |  |
| --- | --- | --- |
| **Start Tag** | **Content** | **End Tag** |
| <p> | This is paragraph content. | </p> |
| <h1> | This is heading content. | </h1> |
| <div> | This is division content. | </div> |
| <br /> |  |  |

So here <p>....</p> is an HTML element, <h1>...</h1> is another HTML element. There are some HTML elements which don't need to be closed, such as <img.../>, <hr /> and <br /> elements. These are known as **void elements**.

HTML documents consist of a tree of these elements and they specify how HTML documents should be built, and what kind of content should be placed in what part of an HTML document.

# *Note: Some HTML elements do not have an end tag.*

# HTML Tag vs. Element:

An HTML element is defined by a *starting tag*. If the element contains other content, it ends with a *closing tag*.

For example <p> is starting tag of a paragraph and </p> is closing tag of the same paragraph but **<p>This is paragraph</p>** is a paragraph element.

# Nested HTML Elements:

It is very much allowed to keep one HTML element inside another HTML element:

HTML elements can be nested (elements can contain elements).

All HTML documents consist of nested HTML elements.

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Nested Elements Example</title>

</head>

<body>

<h1>This is <i>italic</i> heading</h1>

<p>This is <u>underlined</u> paragraph</p>

</body>

</html>

# Don't Forget the End Tag:

Some HTML elements will display correctly, even if you forget the end tag:

Example:

<!DOCTYPE html>

<html>

<body>

<p>This is a paragraph.

<p>This is a paragraph.

</body>

</html>

The example above works in all browsers, because the closing tag is considered optional.

Never rely on this. It might produce unexpected results and/or errors if you forget the end tag.

# Empty HTML Elements:

HTML elements with no content are called empty elements.

<br> is an empty element without a closing tag (the <br> tag defines a line break).

Empty elements can be "closed" in the opening tag like this: <br />.

HTML5 does not require empty elements to be closed. But if you want stricter validation, or you need to make your document readable by XML parsers, you should close all HTML elements.

# HTML Tip: Use Lowercase Tags:

HTML tags are not case sensitive: <P> means the same as <p>.

The HTML5 standard does not require lowercase tags, but we **recommends** lowercase in HTML4, and **demands** lowercase for stricter document types like XHTML.

CHAPTER [5]

# HTML Attributes:

# Attributes provide additional information about HTML elements.

* HTML elements can have **attributes**
* Attributes provide **additional information** about an element
* Attributes are always specified in **the start tag**
* Attributes come in name/value pairs like: **name="value"**

An attribute is used to define the characteristics of an HTML element and is placed inside the element's opening tag. All attributes are made up of two parts: a **name** and a **value**:

* The **name** is the property you want to set. For example, the paragraph <p> element in the example carries an attribute whose name is **align**, which you can use to indicate the alignment of paragraph on the page.
* The **value** is what you want the value of the property to be set and always put within quotations. The below example shows three possible values of align attribute: **left, center** and **right**.

Attribute names and attribute values are case-insensitive. However, the World Wide Web Consortium (W3C) recommends lowercase attributes/attribute values in their HTML 4 recommendation.

### **Example:**

<!DOCTYPE html>

<html>

<head>

<title>Align Attribute Example</title>

</head>

<body>

<p align="left">This is left aligned</p>

<p align="center">This is center aligned</p>

<p align="right">This is right aligned</p>

</body>

</html>

# Core Attributes:

The four core attributes that can be used on the majority of HTML elements (although not all) are:

* id
* title
* class
* style

# The id Attribute:

The **id** attribute of an HTML tag can be used to uniquely identify any element within an HTML page. There are two primary reasons that you might want to use an id attribute on an element:

* If an element carries an id attribute as a unique identifier it is possible to identify just that element and its content.
* If you have two elements of the same name within a Web page (or style sheet), you can use the id attribute to distinguish between elements that have the same name.

# The title Attribute:

The **title** attribute gives a suggested title for the element. They syntax for the**title** attribute is similar as explained for **id** attribute:

The behavior of this attribute will depend upon the element that carries it, although it is often displayed as a tooltip when cursor comes over the element or while the element is loading.

#### EXAMPLE:

<!DOCTYPE html>

<html>

<head>

<title>The title Attribute Example</title>

</head>

<body>

<h3 title="Hello HTML!">Titled Heading Tag Example</h3>

</body>

</html>

# The class Attribute:

The **class** attribute is used to associate an element with a style sheet, and specifies the class of element. You will learn more about the use of the class attribute when you will learn Cascading Style Sheet (CSS). So for now you can avoid it.

The value of the attribute may also be a space-separated list of class names.

### **The style Attribute:**

The style attribute allows you to specify Casecading Style Sheet (CSS) rules within the element.

#### EXAMPLE:

<!DOCTYPE html>

<html>

<head>

<title>The style Attribute</title>

</head>

<body>

<p style="font-family:arial; color:#FF0000;">Some text...</p>

</body>

</html>

# The lang Attribute:

The document language can be declared in the **<html>** tag.

The language is declared in the **lang** attribute.

Declaring a language is important for accessibility applications (screen readers) and search engines:

#### EXAMPLE:

<!DOCTYPE html>  
<html lang="en-US">  
<body>  
  
<h1>My First Heading</h1>  
<p>My first paragraph.</p>  
  
</body>  
</html>

The first two letters specify the language (en). If there is a dialect, use two more letters (US).

# The href Attribute:

HTML links are defined with the **<a>** tag. The link address is specified in the **href** attribute:

#### EXAMPLE:

<!DOCTYPE html>

<html>

<body>

<a href="http:// [www.google.com](http://www.google.com/)">This is a link</a>

</body>

</html>

# Size Attributes:

HTML images are defined with the **<img>** tag.

The filename of the source (**src**), and the size of the image (**width** and **height**) are all provided as **attributes**:

### **Example:**

<!DOCTYPE html>

<html>

<body>

<img src="google.jpg" width="104" height="142">

</body>

</html>

The image size is specified in pixels: width="104" means 104 screen pixels wide.

You will learn more about images and the <img> tag later in this tutorial.

# The alt Attribute:

The **alt** attribute specifies an alternative text to be used, when an HTML element cannot be displayed.

The value of the attribute can be read by "screen readers". This way, someone "listening" to the webpage, i.e. a blind person, can "hear" the element.\

Example:

<!DOCTYPE html>

<html>

<body>

<img src="google.jpg" alt="google.com" width="104" height="142">

</body>

</html>

## The HTML5 standard does not require lower case attribute names.

# We Suggest: Always Use Lowercase Attributes:

The title attribute can be written with upper or lower case like **Title** and/or **TITLE**.

W3C **recommends** lowercase in HTML4, and **demands** lowercase for stricter document types like XHTML.

# Single or Double Quotes?:

Double style quotes are the most common in HTML, but single style can also be used.

In some situations, when the attribute value itself contains double quotes, it is necessary to use single quotes:

<p title='John "ShotGun" Nelson'>

Or vice versa:

<p title="John 'ShotGun' Nelson">

# Chapter Summary:

* All HTML elements can have **attributes**
* The HTML **title** attribute provides additional "tool-tip" information
* The HTML **href** attribute provides address information for links
* The HTML **width** and **height** attributes provide size information for images
* The HTML **alt** attribute provides text for screen readers
* We always use **lowercase** HTML attribute names
* We always **quote** attributes with double quotes.

# HTML Styling:

Every HTML element has a **default style** (background color is white and text color is black).

Changing the default style of an HTML element, can be done with the **style attribute**.

This example changes the default background color from white to lightgrey:

Example:

<!DOCTYPE html>

<html>

<body style="background-color:lightgrey">

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

## The HTML Style Attribute:

The HTML style attribute has the following **syntax**:

style="property:value"

The **property** is a CSS property. The **value** is a CSS value.

# HTML Text Color:

The **color** property defines the text color to be used for an HTML element:

Example:

<!DOCTYPE html>

<html>

<body>

<h1 style="color:blue">This is a heading</h1>

<p style="color:red">This is a paragraph.</p>

</body>

</html>

# HTML Fonts:

The **font-family** property defines the font to be used for an HTML element:

Example:

<!DOCTYPE html>

<html>

<body>

<h1 style="font-family:verdana">This is a heading</h1>

<p style="font-family:courier">This is a paragraph.</p>

</body>

</html>

# HTML Text Size:

The **font-size** property defines the text size to be used for an HTML element:

Example:

<!DOCTYPE html>

<html>

<body>

<h1 style="font-size:300%">This is a heading</h1>

<p style="font-size:160%">This is a paragraph.</p>

</body>

</html>

# HTML Text Alignment:

The **text-align** property defines the horizontal text alignment for an HTML element:

Example:

<!DOCTYPE html>

<html>

<body>

<h1 style="text-align:center">Centered heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

CHAPTER [6]

# HTML Formatting:

# HTML Formatting Elements:

HTML also defines special **elements**, for defining text with a special **meaning**.

HTML uses elements like <b> and <i> for formatting output, like **bold** or *italic* text.

Formatting elements were designed to display special **types of text**:

* Bold text
* Important text
* Italic text
* Emphasized text
* Marked text
* Small text
* Deleted text
* Inserted text
* Subscripts
* Superscripts

# HTML Bold and Strong Formatting:

The HTML **<b>** element defines **bold** text, without any extra importance.

Example:

<!DOCTYPE html>

<html>

<body>

<p>This text is normal.</p>

<p><b>This text is bold.</b></p>

</body>

</html>

The HTML <strong> element defines strong text, with added semantic "strong" importance.

Example:

<!DOCTYPE html>

<html>

<body>

<p>This text is normal.</p>

<p><strong>This text is strong.</strong></p>

</body>

</html>

# Underlined Text:

Anything that appears within <u>...</u> element, is displayed with underline as shown below:

Example:

<!DOCTYPE html>

<html>

<head>

<title>Underlined Text Example</title>

</head>

<body>

<p>The following word uses a <u>underlined</u> typeface.</p>

</body>

</html>

# HTML Small Formatting:

The HTML **<small>** element defines **small** text:

Example:

<!DOCTYPE html>

<html>

<body>

<h2>HTML <small>Small</small> Formatting</h2>

</body>

</html>

# HTML Marked Formatting:

The HTML **<mark>** element defines **marked** or highlighted text:

Example:

<!DOCTYPE html>

<html>

<body>

<h2>HTML <mark>Marked</mark> Formatting</h2>

</body>

</html>

# HTML Deleted Formatting:

The HTML **<del>** element defines **deleted** (removed) text:

Example:

<!DOCTYPE html>

<html>

<body>

<p>The del element represents deleted (removed) text.</p>

<p>My favorite color is <del>blue</del> red.</p>

</body>

</html>

# HTML Inserted Formatting:

The HTML **<ins>** element defines **inserted** (added) text.

Example:

<!DOCTYPE html>

<html>

<body>

<p>The ins element represent inserted (added) text.</p>

<p>My favorite <ins>color</ins> is red.</p>

</body>

</html>

# HTML Subscript Formatting:

The HTML **<sub>** element defines **subscripted** text.

Example:

<!DOCTYPE html>

<html>

<body>

<p>This is <sub>subscripted</sub> text.</p>

</body>

</html>

# HTML Superscript Formatting:

The HTML <sup> element defines superscripted text.

Example:

<!DOCTYPE html>

<html>

<body>

<p>This is <sup>superscripted</sup> text.</p>

</body>

</html>

# Strike Text:

Anything that appears within **<strike>...</strike>** element is displayed with strikethrough, which is a thin line through the text as shown below:

Example:

<!DOCTYPE html>

<html>

<head>

<title>Strike Text Example</title>

</head>

<body>

<p>The following word uses a <strike>strikethrough</strike> typeface.</p>

</body>

</html>

# Monospaced Font:

The content of a **<tt>...</tt>** element is written in monospaced font. Most of the fonts are known as variable-width fonts because different letters are of different widths (for example, the letter 'm' is wider than the letter 'i'). In a monospaced font, however, each letter has the same width.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Monospaced Font Example</title>

</head>

<body>

<p>The following word uses a <tt>monospaced</tt> typeface.</p>

</body>

</html>

# Emphasized Text:

Anything that appears within **<em>...</em>** element is displayed as emphasized text.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Emphasized Text Example</title>

</head>

<body>

<p>The following word uses a <em>emphasized</em> typeface.</p>

</body>

</html>

CHAPTER [7]

# HTML Quotation:

# HTML <q> for Short Quotations:

The HTML **<q>** element defines a short quotation.

Browsers usually insert quotation marks around the <q> element.

Example:

<!DOCTYPE html>

<html>

<body>

<p>Browsers usually insert quotation marks around the q element.</p>

<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>

</body>

</html>

# HTML <blockquote> for Long Quotations:

The HTML **<blockquote>** element defines a quoted section.

Browsers usually indent <blockquote> elements.

Example:

<!DOCTYPE html>

<html>

<body>

<p>Browsers usually indent blockquote elements.</p>

<blockquote cite="http://www.worldwildlife.org/who/index.html">

For 50 years, WWF has been protecting the future of nature.

The world's leading conservation organization,

WWF works in 100 countries and is supported by

1.2 million members in the United States and

close to 5 million globally.

</blockquote>

</body>

</html>

# HTML <abbr> for Abbreviations:

The HTML **<abbr>** element defines an abbreviation or an acronym.

Marking abbreviations can give useful information to browsers, translation systems and search-engines.

Example:

<!DOCTYPE html>

<html>

<body>

<p>The <abbr title="World Health Organization">WHO</abbr> was founded in 1948.</p>

<p>Marking up abbreviations can give useful information to browsers, translation systems and search-engines.</p>

</body>

</html>

# Text Direction:

The **<bdo>...</bdo>** element stands for Bi-Directional Override and it is used to override the current text direction.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Text Direction Example</title>

</head>

<body>

<p>This text will go left to right.</p>

<p><bdo dir="rtl">This text will go right to left.</bdo></p>

</body>

</html>

# Text Citations:

If you are quoting a text, you can indicate the source placing it between an opening **<cite>** tag and closing **</cite>** tag

As you would expect in a print publication, the content of the <cite> element is rendered in italicized text by default.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Citations Example</title>

</head>

<body>

<p>This HTML tutorial is derived from <cite>W3C Standard for HTML</cite>.</p>

</body>

</html>

# HTML <address> for Contact Information:

The HTML **<address>** element defines contact information (author/owner) of a document or article.

The <address> element is usually displayed in italic. Most browsers will add a line break before and after the element.

Example:

<!DOCTYPE html>

<html>

<body>

<p>The HTML address element defines contact information (author/owner) of a document or article.</p>

<address>Written by Jon Doe.<br>

Visit us at:<br>

Example.com<br>

Box 564, Disneyland<br>

USA

</address>

</body>

</html>

# HTML Computer Code Formatting:

Normally, HTML uses **variable** letter size, and variable letter spacing.

This is not wanted when displaying examples of **computer code**.

The **<kbd>**, **<samp>**, and **<code>** elements all support **fixed** letter size and spacing.

# HTML Code Formatting:

The HTML **<code>** element defines **programming code**:

Any programming code to appear on a Web page should be placed inside**<code>...</code>** tags. Usually the content of the <code> element is presented in a monospaced font, just like the code in most programming books.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Computer Code Example</title>

</head>

<body>

<code>

var person = {firstName:"John", lastName:"Doe", age:50}

</code>

</body>

</html>

Note: The <code> element does not preserve extra whitespace and line-breaks:

# Keyboard Text:

When you are talking about computers, if you want to tell a reader to enter some text, you can use the **<kbd>...</kbd>** element to indicate what should be typed in, as in this example.

Example:

<!DOCTYPE html>

<html>

<body style="font-size:16px">

<p>The kbd element represents keyboard input:</p>

<p><kbd>File | Open...</kbd></p>

</body>

</html>

# HTML Sample Formatting:

The HTML **<samp>** element defines a **computer output**:

The **<samp>...</samp>** element indicates sample output from a program, and script etc. Again, it is mainly used when documenting programming or coding concepts.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Program Output Example</title>

</head>

<body>

<p>Result produced by the program is <samp>Hello World!</samp></p>

</body>

</html>

# Programming Variables:

This element is usually used in conjunction with the **<pre>** and **<code>**elements to indicate that the content of that element is a variable.

Example:

<!DOCTYPE html>

<html>

<body>

<p>The code element does not preserve whitespace and line-breaks.</p>

<p>To fix this, you can wrap the code in a pre element:</p>

<code>

<pre>

var person = {

firstName:"John",

lastName:"Doe",

age:50,

eyeColor:"blue"

}

</pre>

</code>

</body>

</html>

# HTML Variable Formatting:

The HTML **<var>** element defines a **mathematical variable**:

Example:

<!DOCTYPE html>

<html>

<body style="font-size:16px">

<p>Einstein wrote:</p>

<p><var>E</var> = <var>m</var> <var>c</var><sup>2</sup></p>

</body>

</html>

CHAPTER [8]

# HTML Comments:

Comment tags <!-- and --> are used to insert comments in HTML.

Comment is a piece of code which is ignored by any web browser. It is a good practice to add comments into your HTML code, especially in complex documents, to indicate sections of a document, and any other notes to anyone looking at the code. Comments help you and others understand your code and increases code readability.

HTML comments are placed in between **<!-- ... -->** tags. So any content placed with-in <!-- ... --> tags will be treated as comment and will be completely ignored by the browser.

Example:

<!DOCTYPE html><html>

<head>

<title>Multiline Comments</title>

</head>

<body>

<!--

This is a multiline comment and it can

span through as many as lines you like.

-->

<p>Document content goes here.....</p>

</body>

</html>

# Conditional Comments:

Conditional comments only work in Internet Explorer (IE) on Windows but they are ignored by other browsers. They are supported from Explorer 5 onwards, and you can use them to give conditional instructions to different versions of IE.

Example:

<!DOCTYPE html><html>

<head>

<title>Conditional Comments</title>

<!--[if IE 6]>

Special instructions for IE 6 here

<![endif]-->

</head>

<body>

<p>Document content goes here.....</p>

</body>

</html>

You will come across a situation where you will need to apply a different style sheet based on different versions of Internet Explorer, in such situation conditional comments will be helpful.

# Using Comment Tag:

There are few browsers that support <comment> tag to comment a part of HTML code.

Example:

<!DOCTYPE html><html>

<head>

<title>Using Comment Tag</title>

</head>

<body>

<p>This is <comment>not</comment> Internet Explorer.</p>

</body>

</html>

Note: If you are using IE then it will produce following result:

This is Internet Explorer.

Note: But if you are not using IE, then it will produce following result:

This is not Internet Explorer.

# Commenting Script Code:

Though you will learn Javascript with HTML, in a separate tutorial, but here you must make a note that if you are using Java Script or VB Script in your HTML code then it is recommended to put that script code inside proper HTML comments so that old browsers can work properly.

Example:

<!DOCTYPE html><html>

<head>

<title>Commenting Script Code</title>

<script>

<!—

document.write("Hello World!")

//-->

</script>

</head>

<body>

<p>Hello , World!</p>

</body>

</html>

# Commenting Style Sheets:

Though you will learn using style sheets with HTML in a separate tutorial, but here you must make a note that if you are using Casecading Style Sheet (CSS) in your HTML code then it is recommended to put that style sheet code inside proper HTML comments so that old browsers can work properly.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Commenting Style Sheets</title>

<style>

<!--

.example {

border:1px solid #4a7d49;

} //-->

</style>

</head>

<body>

<div class="example">Hello , World!</div>

</body>

</html>

CHAPTER [9]

# HTML Style Sheet:

Cascading Style Sheets (CSS) describe how documents are presented on screens, in print, or perhaps how they are pronounced. W3C has actively promoted the use of style sheets on the Web since the Consortium was founded in 1994.

Cascading Style Sheets (CSS) provide easy and effective alternatives to specify various attributes for the HTML tags. Using CSS, you can specify a number of style properties for a given HTML element. Each property has a name and a value, separated by a colon (:). Each property declaration is separated by a semi-colon (;).

# CSS Syntax:

CSS styling has the following syntax:

element { property:value; property:value }

The **element** is an HTML element name. The ***property*** is a CSS property. The ***value*** is a CSS value.

Multiple styles are separated with semicolon.

# Styling HTML with CSS:

CSS stands for **C**ascading **S**tyle **S**heets

Styling can be added to HTML elements in 3 ways:

Inline - using a **style attribute** in HTML elements

Internal - using a **<style> element** in the HTML <head> section

External - using one or more **external CSS files**

The most common way to add styling, is to keep the styles in separate CSS files. But, in this tutorial, we use internal styling, because it is easier to demonstrate, and easier for you to try it yourself.

You can use CSS in three ways in your HTML document:

**External Style Sheet** - Define style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.

**Internal Style Sheet** - Define style sheet rules in header section of the HTML document using <style> tag.

**Inline Style Sheet** - Define style sheet rules directly along-with the HTML elements using **style** attribute.

Let's see all the three cases one by one with the help of suitable examples.

# E**xternal** Style Sheet:

If you need to use your style sheet to various pages, then its always recommended to define a common style sheet in a separate file. A cascading style sheet file will have extension as **.css** and it will be included in HTML files using <link> tag.

Example:

Consider we define a style sheet file **style.css** which has following rules:

.red{

color: red;

}

.thick{

font-size:20px;

}

.green{

color:green;

}

Here we defined three CSS rules which will be applicable to three different classes defined for the HTML tags. I suggest you should not bother about how these rules are being defined because you will learn them while studying CSS. Now let's make use of the above external CSS file in our following HTML document:

<!DOCTYPE html>

<html>

<head>

<title>HTML External CSS</title>

<link rel="stylesheet" type="text/css" href="/html/style.css">

</head>

<body>

<p class="red">This is red</p>

<p class="thick">This is thick</p>

<p class="green">This is green</p>

<p class="thick green">This is thick and green</p>

</body>

</html>

# Internal Style Sheet:

If you want to apply Style Sheet rules to a single document only then you can include those rules in header section of the HTML document using <style> tag.

Rules defined in internal style sheet overrides the rules defined in an external CSS file.

Example:

Let's re-write above example once again, but here we will write style sheet rules in the same HTML document using <style> tag:

<!DOCTYPE html>

<html>

<head>

<title>HTML Internal CSS</title>

<style type="text/css">

.red{

color: red;

}

.thick{

font-size:20px;

}

.green{

color:green;

}

</style>

</head>

<body>

<p class="red">This is red</p>

<p class="thick">This is thick</p>

<p class="green">This is green</p>

<p class="thick green">This is thick and green</p>

</body>

</html>

# Inline Style Sheet:

You can apply style sheet rules directly to any HTML element using **style**attribute of the relevant tag. This should be done only when you are interested to make a particular change in any HTML element only.

Rules defined inline with the element overrides the rules defined in an external CSS file as well as the rules defined in <style> element.

Example:

Let's re-write above example once again, but here we will write style sheet rules along with the HTML elements using **style** attribute of those elements.

<!DOCTYPE html>

<html>

<head>

<title>HTML Inline CSS</title>

</head>

<body>

<p style="color:red;">This is red</p>

<p style="font-size:20px;">This is thick</p>

<p style="color:green;">This is green</p>

<p style="color:green;font-size:20px;">This is thick and green</p>

</body>

</html>

# CSS Fonts:

The CSS **color** property defines the text color to be used for the HTML element.

The CSS **font-family** property defines the font to be used for the HTML element.

The CSS **font-size** property defines the text size to be used for the HTML element.

Example:

<!DOCTYPE html>

<html>

<head>

<style>

h1 {

color:blue;

font-family:verdana;

font-size:300%;

}

p {

color:red;

font-family:courier;

font-size:160%;

}

</style>

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

# The CSS Box Model:

Every HTML element has a box around it, even if you cannot see it.

The CSS **border** property defines a visible border around an HTML element:

<!DOCTYPE html>

<html>

<head>

<style>

p {

border:1px solid grey;

}

</style>

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

</body>

</html>

The CSS **padding** property defines a padding (space) inside the border:

<!DOCTYPE html>

<html>

<head>

<style>

p {

border:1px solid grey;

padding:10px;

}

</style>

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

</body>

</html>

The CSS **margin** property defines a margin (space) outside the border:

<!DOCTYPE html>

<html>

<head>

<style>

p {

border:1px solid grey;

padding:10px;

margin:30px;

}

</style>

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

</body>

</html>

# The id Attribute:

All the examples above use CSS to style HTML elements in a general way.

To define a special style for one special element, first add an id attribute to the element:

Example:

<p id="p01">I am different</p>

then define a different style for the (identified) element:

Example:

<!DOCTYPE html>

<html>

<head>

<style>

p#p01 {

color: blue;

}

</style>

</head>

<body>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

<p id="p01">I am different.</p>

</body>

</html>

# The class Attribute:

To define a style for a special type (class) of elements, add a class attribute to the element:

Example:

<p class="error">I am different</p>

Now you can define a different style for all elements with the specified class:

Example:

<!DOCTYPE html>

<html>

<head>

<style>

p.error {

color:red;

}

</style>

</head>

<body>

<p>This is a paragraph.</p>

<p>This is a paragraph.</p>

<p class="error">I am different.</p>

<p>This is a paragraph.</p>

<p class="error">I am different too.</p>

</body>

</html>

# Deprecated Tags and Attributes in HTML5:

In older HTML versions, several tags and attributes were used to style documents.

These tags and attributes are not supported in HTML5!

Avoid using the <font>, <center>, and <strike> elements.

Avoid using the color and bgcolor attributes.

CHAPTER [10]

# HTML Links:

# H**TML Links – Hyperlinks:**

HTML links are hyperlinks.

A hyperlink is a text or an image you can click on, and jump to another document.

# HTML Links – Syntax:

In HTML, links are defined with the **<a>** tag:

<a href="*url*">*link text*</a>

Example:

<!DOCTYPE html>

<html>

<body>

<p><a href="http:// [www.google.com](http://www.google.com/)/html/">Visit our HTML tutorial</a></p>

</body>

</html>

The **href** attribute specifies the destination address (http:// [www.google.com](http://www.google.com/)/html/)

The **link text** is the visible part (Visit our HTML tutorial).

Clicking on the link text, will send you to the specified address.

# Local Links:

The example above used an absolute URL (A full web address).

A local link (link to the same web site) is specified with a relative URL (without http://www....).

# Example:

<!DOCTYPE html>

<html>

<body>

<p><a href="html\_images.asp">HTML Images</a> is a link to a page on this website.</p>

<p><a href="http://www. Google.org/">W3C</a> is a link to a website on the World Wide Web.</p>

</body>

</html>

# HTML Links – Colors:

When you move the mouse over a link, two things will normally happen:

* The mouse arrow will turn into a little hand
* The color of the link element will change

By default, a link will appear like this (in all browsers):

* An unvisited link is underlined and blue
* A visited link is underlined and purple
* An active link is underlined and red

You can change the default colors, by using styles:

Example:

<!DOCTYPE html>

<html>

<head>

<style>

a:link {

color: green;

background-color: transparent;

text-decoration: none;

}

a:visited {

color: pink;

background-color: transparent;

text-decoration: none;

}

a:hover {

color: red;

background-color: transparent;

text-decoration: underline;

}

a:active {

color: yellow;

background-color: transparent;

text-decoration: underline;

}

</style>

</head>

<body>

<p>You can change the default colors of links</p>

<a href="html\_images.asp" target="\_blank">HTML Images</a>

</body>

</html>

# HTML Links - The target Attribute:

The **target** attribute specifies where to open the linked document.

This example will open the linked document in a new browser window or in a new tab:

Example:

<a href="http:// [www.google.com](http://www.google.com/)/" target="\_blank">Visit Google</a>

|  |  |
| --- | --- |
| Target Value | Description |
|  |  |
| \_blank | Opens the linked document in a new window or tab |
| \_self | Opens the linked document in the same frame as it was clicked (this is default) |
| \_parent | Opens the linked document in the parent frame |
| \_top | Opens the linked document in the full body of the window |
| *framename* | Opens the linked document in a named frame |

If your webpage is locked in a frame, you can use target="\_top" to break out of the frame:

Example:

<!DOCTYPE html>

<html>

<body>

<p>Locked in a frame? <a href="http:// [www.google.com](http://www.google.com/)/html/" target="\_top">Click here!</a></p>

</body>

</html>

# HTML Links - Image as Link:

It is common to use images as links:

Example:

<!DOCTYPE html>

<html>

<body>

<p>The image is a link. You can click on it.</p>

<a href="default.asp">

<img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;border:0">

</a>

<p>We have added "border:0" to prevent IE9 (and earlier) from displaying a border around the image.</p>

</body>

</html>

# HTML Links - Create a Bookmark:

HTML bookmarks are used to allow readers to jump to specific parts of a Web page.

Bookmarks are practical if your website has long pages.

To make a bookmark, you must first create the bookmark, and then add a link to it.

When the link is clicked, the page will scroll to the location with the bookmark.

Example:

First, create a bookmark with the id attribute:

<h2 id="tips">Useful Tips Section</h2>

Then, add a link to the bookmark ("Useful Tips Section"), from within the same page:

<a href="#tips">Visit the Useful Tips Section</a>

Or, add a link to the bookmark ("Useful Tips Section"), from another page:

Example:

<!DOCTYPE html>

<html>

<body>

<p><a href="#C4">Jump to Chapter 4</a></p>

<h2>Chapter 1</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 2</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 3</h2>

<p>This chapter explains ba bla bla</p>

<h2 id="C4">Chapter 4</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 5</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 6</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 7</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 8</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 9</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 10</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 11</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 12</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 13</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 14</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 15</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 16</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 17</h2>

<p>This chapter explains ba bla bla</p>

</body>

</html>

CHAPTER [11]

# HTML Images:

Images are very important to beautify as well as to depict many complex concepts in simple way on your web page. This tutorial will take you through simple steps to use images in your web pages.

# HTML Images Syntax:

In HTML, images are defined with the **<img>** tag.

The <img> tag is empty, it contains attributes only, and does not have a closing tag.

The src attribute specifies the URL (web address) of the image:

<img src="*url*" alt="*some\_text*">

# Insert Image:

You can insert any image in your web page by using **<img>** tag. Following is the simple syntax to use this tag.

<img src="Image URL" ... attributes-list/>

The <img> tag is an empty tag, which means that it can contain only list of attributes and it has no closing tag.

Example:

To try following example, let's keep our HTML file test.htm and image file test.png in the same directory:

<!DOCTYPE html>

<html>

<head>

<title>Using Image in Webpage</title>

</head>

<body>

<p>Simple Image Insert</p>

<img src="/html/images/test.png" alt="Test Image" />

</body>

</html>

You can use PNG, JPEG or GIF image file based on your comfort but make sure you specify correct image file name in **src** attribute. Image name is always case sensitive.

The **alt** attribute is a mandatory attribute which specifies an alternate text for an image, if the image cannot be displayed.

# Set Image Location:

Usually we keep our all the images in a separate directory. So let's keep HTML file test.htm in our home directory and create a subdirectory **images** inside the home directory where we will keep our image test.png.

Example:

Assuming our image location is "/html/image/test.png", try the following example:

<!DOCTYPE html>

<html>

<head>

<title>Using Image in Webpage</title>

</head>

<body>

<p>Simple Image Insert</p>

<img src="/html/images/test.png" alt="Test Image" />

</body>

</html>

# Set Image Width/Height:

You can set image width and height based on your requirement using widthand height attributes. You can specify width and height of the image in terms of either pixels or percentage of its actual size.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Set Image Width and Height</title>

</head>

<body>

<p>Setting image width and height</p>

<img src="/html/images/test.png" alt="Test Image" width="150" height="100"/>

</body>

</html>

# Set Image Border:

By default image will have a border around it, you can specify border thickness in terms of pixels using **border** attribute. A thickness of 0 means, no border around the picture.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Set Image Border</title>

</head>

<body>

<p>Setting image Border</p>

<img src="/html/images/test.png" alt="Test Image" border="3"/>

</body>

</html>

# Set Image Alignment:

By default image will align at the left side of the page, but you can use **align**attribute to set it in the center or right.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Set Image Alignment</title>

</head>

<body>

<p>Setting image Alignment</p>

<img src="/html/images/test.png" alt="Test Image" border="3" align="right"/>

</body>

</html>

# Images on Another Server:

Some web sites store their images on image servers.

Actually, you can access images from any web address in the world:

Example:

<!DOCTYPE html>

<html>

<body>

<img src="http:// [www.google.com](http://www.google.com/)/images/Google\_green.jpg" alt="[google.com](http://www.google.com/)" style="width:104px;height:142px;">

</body>

</html>

# Using an Image as a Link:

 To use an image as a link, simply nest the <img> tag inside the <a> tag:

Example:

<!DOCTYPE html>

<html>

<body>

<p>The image is a link. You can click on it.</p>

<a href="default.asp">

<img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;border:0;">

</a>

<p>Add "border:0;" to prevent IE9 (and earlier) from displaying a border around the image.</p>

</body>

</html>

# Image Maps:

Use the <map> tag to define an image-map. An image-map is an image with clickable areas.

The name attribute of the <map> tag is associated with the <img>'s usemap attribute and creates a relationship between the image and the map.

The <map> tag contains a number of <area> tags, that defines the clickable areas in the image-map:

Example:

<!DOCTYPE html>

<html>

<body>

<p>Click on the sun or on one of the planets to watch it closer:</p>

<img src="planets.gif" alt="Planets" usemap="#planetmap" style="width:145px;height:126px;">

<map name="planetmap">

<area shape="rect" coords="0,0,82,126" alt="Sun" href="sun.htm">

<area shape="circle" coords="90,58,3" alt="Mercury" href="mercur.htm">

<area shape="circle" coords="124,58,8" alt="Venus" href="venus.htm">

</map>

</body>

</html>

# Image Floating:

Use the CSS float property to let the image float.

The image can float to the right or to the left of a text:

Example:

<!DOCTYPE html>

<html>

<body>

<p><strong>Float the image to the right:</strong></p>

<p>

<img src="smiley.gif" alt="Smiley face" style="float:right;width:42px;height:42px;">

A paragraph with a floating image. A paragraph with a floating image. A paragraph with a floating image.

</p>

<p><strong>Float the image to the left:</strong></p>

<p>

<img src="smiley.gif" alt="Smiley face" style="float:left;width:42px;height:42px;">

A paragraph with a floating image. A paragraph with a floating image. A paragraph with a floating image.

</p>

<p>Please use the CSS float property. The align attribute is deprecated in HTML 4, and not supported in HTML5.</p>

</body>

</html>

CHAPTER [12]

# HTML Tables:

The HTML tables allow web authors to arrange data like text, images, links, other tables, etc. into rows and columns of cells.

The HTML tables are created using the **<table>** tag in which the **<tr>** tag is used to create table rows and **<td>** tag is used to create data cells.

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Tables</title>

</head>

<body>

<table border="1">

<tr>

<td>Row 1, Column 1</td>

<td>Row 1, Column 2</td>

</tr>

<tr>

<td>Row 2, Column 1</td>

<td>Row 2, Column 2</td>

</tr>

</table>

</body>

</html>

Here **border** is an attribute of <table> tag and it is used to put a border across all the cells. If you do not need a border then you can use border="0".

# Defining HTML Tables:

Example:

<!DOCTYPE html>

<html>

<body>

<table style="width:100%">

<tr>

<td>Jill</td>

<td>Smith</td>

<td>50</td>

</tr>

<tr>

<td>Eve</td>

<td>Jackson</td>

<td>94</td>

</tr>

<tr>

<td>John</td>

<td>Doe</td>

<td>80</td>

</tr>

</table>

</body>

</html>

Example explained:

Tables are defined with the **<table>** tag.

Tables are divided into **table rows** with the **<tr>** tag.

Table rows are divided into **table data** with the **<td>** tag.

A table row can also be divided into **table headings** with the **<th>** tag.

# Table Heading:

Table heading can be defined using **<th>** tag. This tag will be put to replace <td> tag, which is used to represent actual data cell. Normally you will put your top row as table heading as shown below, otherwise you can use <th> element in any row.

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Header</title>

</head>

<body>

<table border="1">

<tr>

<th>Name</th>

<th>Salary</th>

</tr>

<tr>

<td>Ramesh Raman</td>

<td>5000</td>

</tr>

<tr>

<td>Shabbir Hussein</td>

<td>7000</td>

</tr>

</table>

</body>

</html>

# Cellpadding and Cellspacing Attributes:

There are two attribiutes called *cellpadding* and *cellspacing* which you will use to adjust the white space in your table cells. The cellspacing attribute defines the width of the border, while cellpadding represents the distance between cell borders and the content within a cell.

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Cellpadding</title>

</head>

<body>

<table border="1" cellpadding="5" cellspacing="5">

<tr>

<th>Name</th>

<th>Salary</th>

</tr>

<tr>

<td>Ramesh Raman</td>

<td>5000</td>

</tr>

<tr>

<td>Shabbir Hussein</td>

<td>7000</td>

</tr>

</table>

</body>

</html>

# Colspan and Rowspan Attributes:

You will use **colspan** attribute if you want to merge two or more columns into a single column. Similar way you will use **rowspan** if you want to merge two or more rows.

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Colspan/Rowspan</title>

</head>

<body>

<table border="1">

<tr>

<th>Column 1</th>

<th>Column 2</th>

<th>Column 3</th>

</tr>

<tr><td rowspan="2">Row 1 Cell 1</td><td>Row 1 Cell 2</td><td>Row 1 Cell 3</td></tr>

<tr><td>Row 2 Cell 2</td><td>Row 2 Cell 3</td></tr>

<tr><td colspan="3">Row 3 Cell 1</td></tr>

</table>

</body>

</html>

# Tables Backgrounds:

You can set table background using one of the following two ways:

* **bgcolor** attribute - You can set background color for whole table or just for one cell.
* **background** attribute - You can set background image for whole table or just for one cell.

You can also set border color also using **bordercolor** attribute.

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Background</title>

</head>

<body>

<table border="1" bordercolor="green" bgcolor="yellow">

<tr>

<th>Column 1</th>

<th>Column 2</th>

<th>Column 3</th>

</tr>

<tr><td rowspan="2">Row 1 Cell 1</td><td>Row 1 Cell 2</td><td>Row 1 Cell 3</td></tr>

<tr><td>Row 2 Cell 2</td><td>Row 2 Cell 3</td></tr>

<tr><td colspan="3">Row 3 Cell 1</td></tr>

</table>

</body>

</html>

Here is an example of using **background** attribute. Here we will use an image available in /images directory.

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Background</title>

</head>

<body>

<table border="1" bordercolor="green" background="/images/test.png">

<tr>

<th>Column 1</th>

<th>Column 2</th>

<th>Column 3</th>

</tr>

<tr><td rowspan="2">Row 1 Cell 1</td><td>Row 1 Cell 2</td><td>Row 1 Cell 3</td></tr>

<tr><td>Row 2 Cell 2</td><td>Row 2 Cell 3</td></tr>

<tr><td colspan="3">Row 3 Cell 1</td></tr>

</table>

</body>

</html>

# Table Height and Width:

You can set a table width and height using **width** and **height** attrubutes. You can specify table width or height in terms of pixels or in terms of percentage of available screen area.

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Width/Height</title>

</head>

<body>

<table border="1" width="400" height="150">

<tr>

<td>Row 1, Column 1</td>

<td>Row 1, Column 2</td>

</tr>

<tr>

<td>Row 2, Column 1</td>

<td>Row 2, Column 2</td>

</tr>

</table>

</body>

</html>

# Table Caption:

The **caption** tag will serve as a title or explanation for the table and it shows up at the top of the table. This tag is deprecated in newer version of HTML/XHTML.

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Caption</title>

</head>

<body>

<table border="1" width="100%">

<caption>This is the caption</caption>

<tr>

<td>row 1, column 1</td><td>row 1, columnn 2</td>

</tr>

<tr>

<td>row 2, column 1</td><td>row 2, columnn 2</td>

</tr>

</table>

</body>

</html>

# Table Header, Body, and Footer:

Tables can be divided into three portions: a header, a body, and a foot. The head and foot are rather similar to headers and footers in a word-processed document that remain the same for every page, while the body is the main content holder of the table.

The three elements for separating the head, body, and foot of a table are:

* **<thead> -**to create a separate table header.
* **<tbody> -**to indicate the main body of the table.
* **<tfoot> -**to create a separate table footer.

A table may contain several <tbody> elements to indicate different *pages* or groups of data. But it is notable that <thead> and <tfoot> tags should appear before <tbody>

Example:

<!DOCTYPE html>

<html>

<head>

<title>HTML Table</title>

</head>

<body>

<table border="1" width="100%">

<thead>

<tr>

<td colspan="4">This is the head of the table</td>

</tr>

</thead>

<tfoot>

<tr>

<td colspan="4">This is the foot of the table</td>

</tr>

</tfoot>

<tbody>

<tr>

<td>Cell 1</td>

<td>Cell 2</td>

<td>Cell 3</td>

<td>Cell 4</td>

</tr>

</tbody>

</table>

</body>

</html>

# Nested Tables:

You can use one table inside another table. Not only tables you can use almost all the tags inside table data tag <td>.

Example:

Following is the example of using another table and other tags inside a table cell.

<!DOCTYPE html>

<html>

<head>

<title>HTML Table</title>

</head>

<body>

<table border="1" width="100%">

<tr>

<td>

<table border="1" width="100%">

<tr>

<th>Name</th>

<th>Salary</th>

</tr>

<tr>

<td>Ramesh Raman</td>

<td>5000</td>

</tr>

<tr>

<td>Shabbir Hussein</td>

<td>7000</td>

</tr>

</table>

</td>

</tr>

</table>

</body>

</html>

CHAPTER [13]

# HTML Lists:

HTML offers web authors three ways for specifying lists of information. All lists must contain one or more list elements. Lists may contain:

* **<ul>** - An unordered list. This will list items using plain bullets.
* **<ol>** - An ordered list. This will use different schemes of numbers to list your items.
* **<dl>** - A definition list. This arranges your items in the same way as they are arranged in a dictionary.

Unordered lists and ordered lists are commonly used in HTML:

Unordered List:

* The first item
* The second item
* The third item
* The fourth item

Ordered List:

1. The first item
2. The second item
3. The third item
4. The fourth item

# Unordered HTML Lists:

An unordered list starts with the **<ul>** tag. Each list item starts with the **<li>** tag.

The list items will be marked with bullets (small black circles):

Example:

<!DOCTYPE html>

<html>

<body>

<h2>Unordered List with Default Bullets</h2>

<ul>

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ul>

</body>

</html>

# Unordered HTML Lists - The Style Attribute:

A **style** attribute can be added to an **unordered list**, to define the style of the marker:

|  |  |
| --- | --- |
| **STYLE** | **DESCRIPTION** |
| list-style-type:disc | The list items will be marked with bullets (default) |
| list-style-type:circle | The list items will be marked with circle |
| list-style-type:square | The list items will be marked with square |
| list-style-type:none | The list items will not be marked |

# Disc:

<!DOCTYPE html>

<html>

<body>

<h2>Unordered List with Disc Bullets</h2>

<ul style="list-style-type:disc">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ul>

</body>

</html>

# Circle:

<!DOCTYPE html>

<html>

<body>

<h2>Unordered List with Circle Bullets</h2>

<ul style="list-style-type:circle">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ul>

</body>

</html>

# Square:

<!DOCTYPE html>

<html>

<body>

<h2>Unordered List with Square Bullets</h2>

<ul style="list-style-type:square">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ul>

</body>

</html>

# None:

<!DOCTYPE html>

<html>

<body>

<h2>Unordered List without Bullets</h2>

<ul style="list-style-type:none">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ul>

</body>

</html>

# Ordered HTML Lists:

An ordered list starts with the **<ol>** tag. Each list item starts with the **<li>** tag.

The list items will be marked with numbers:

Example:

<!DOCTYPE html>

<html>

<body>

<h2>Ordered List</h2>

<ol>

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

</body>

</html>

# Ordered HTML Lists - The Type Attribute:

A **type** attribute can be added to an **ordered list**, to define the type of the marker:

A **style** attribute can be added to an **unordered list**, to define the style of the marker:

|  |  |
| --- | --- |
| **TYPE** | **DESCRIPTION** |
| type="1" | The list items will be numbered with numbers (default) |
| type="A" | The list items will be numbered with uppercase letters |
| type="a" | The list items will be numbered with lowercase letters |
| type="I" | The list items will be numbered with uppercase roman numbers |
| type="i" | The list items will be numbered with lowercase roman numbers |

# Numbers:

<!DOCTYPE html>

<html>

<body>

<h2>Ordered List with Numbers</h2>

<ol type="1">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

</body>

</html>

# Uppercase Letters:

<!DOCTYPE html>

<html>

<body>

<h2>Ordered List with Letters</h2>

<ol type="A">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

</body>

</html>

# Lowercase Letters:

<!DOCTYPE html>

<html>

<body>

<h2>Ordered List with Lowercase Letters</h2>

<ol type="a">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

</body>

</html>

# Uppercase Roman Numbers:

<!DOCTYPE html>

<html>

<body>

<h2>Ordered List with Roman Numbers</h2>

<ol type="I">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

</body>

</html>

# Lowercase Roman Numbers:

<!DOCTYPE html>

<html>

<body>

<h2>Ordered List with Lowercase Roman Numbers</h2>

<ol type="i">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

</body>

</html>

# HTML Description Lists:

HTML also supports description lists.

A description list is a list of terms, with a description of each term.

The **<dl>** tag defines the description list, the **<dt>** tag defines the term (name), and the **<dd>** tag describes each term:

Example:

<!DOCTYPE html>

<html>

<body>

<h2>A Description List</h2>

<dl>

<dt>Coffee</dt>

<dd>- black hot drink</dd>

<dt>Milk</dt>

<dd>- white cold drink</dd>

</dl>

</body>

</html>

# Nested HTML Lists:

List can be nested (lists inside lists):

Example:

<!DOCTYPE html>

<html>

<body>

<h2>A Nested List</h2>

<ul>

<li>Coffee</li>

<li>Tea

<ul>

<li>Black tea</li>

<li>Green tea</li>

</ul>

</li>

<li>Milk</li>

</ul>

</body>

</html>

# Horizontal Lists:

HTML lists can be styled in many different ways with CSS.

One popular way, is to style a list to be displayed horizontally:

Example:

<!DOCTYPE html>

<html>

<head>

<style>

ul#menu li {

display:inline;

}

</style>

</head>

<body>

<h2>Horizontal List</h2>

<ul id="menu">

<li>HTML</li>

<li>CSS</li>

<li>JavaScript</li>

<li>PHP</li>

</ul>

</body>

</html>

With a little extra style, you can make it look like a menu:

<!DOCTYPE html>

<html>

<head>

<style>

ul#menu {

padding: 0;

}

ul#menu li {

display: inline;

}

ul#menu li a {

background-color: black;

color: white;

padding: 10px 20px;

text-decoration: none;

border-radius: 4px 4px 0 0;

}

ul#menu li a:hover {

background-color: orange;

}

</style>

</head>

<body>

<h2>Horizontal List</h2>

<ul id="menu">

<li><a href="/html/default.asp">HTML</a></li>

<li><a href="/css/default.asp">CSS</a></li>

<li><a href="/js/default.asp">JavaScript</a></li>

<li><a href="/php/default.asp">PHP</a></li>

</ul>

</body>

</html>

CHAPTER [14]

# HTML Blocks:

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline.

All the HTML elements can be categorized into two categories (a) Block Level Elements (b) Inline Elements.

# Block Elements:

Block elements appear on the screen as if they have a line break before and after them. For example the <p>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, <ul>, <ol>, <dl>, <pre>, <hr />, <blockquote>, and <address> elements are all block level elements. They all start on their own new line, and anything that follows them appears on its own new line.

# Inline Elements:

Inline elements, on the other hand, can appear within sentences and do not have to appear on a new line of their own. The <b>, <i>, <u>, <em>, <strong>, <sup>, <sub>, <big>, <small>, <li>, <ins>, <del>, <code>, <cite>, <dfn>, <kbd>, and <var> elements are all inline elements.

# Grouping HTML Elements:

There are two important tags which we use very frequently to group various other HTML tags (i) <div> tag and (ii) <span> tag

# The <div> tag:

This is the very important block level tag which plays a big role in grouping various other HTML tags and applying CSS on group of elements. Even now <div> tag can be used to create webpage layout where we define different parts ( Left, Right, Top etc) of the page using <div> tag. This tag does not provide any visual change on the block but this has more meaning when it is used with CSS.

Example:

Following is a simple example of <div> tag. We will learn Cascading Style Sheet (CSS) in a separate chapter but we used it here to show the usage of <div> tag:

<!DOCTYPE html>

<html>

<head>

<title>HTML div Tag</title>

</head>

<body>

<!-- First group of tags -->

<div style="color:red">

<h4>This is first group</h4>

<p>Following is a list of vegetables</p>

<ul>

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ul>

</div>

<!-- Second group of tags -->

<div style="color:green">

<h4>This is second group</h4>

<p>Following is a list of fruits</p>

<ul>

<li>Apple</li>

<li>Banana</li>

<li>Mango</li>

<li>Strawberry</li>

</ul>

</div>

</body>

</html>

# The <span> tag:

The HTML <span> is an inline element and it can be used to group inline-elements in an HTML document. This tag also does not provide any visual change on the block but has more meaning when it is used with CSS.

The difference between the <span> tag and the <div> tag is that the <span> tag is used with inline elements where as the <div> tag is used with block-level elements.

Example:

Following is a simple example of <span> tag. We will learn Cascading Style Sheet (CSS) in a separate chapter but we used it here to show the usage of <span> tag:

<!DOCTYPE html>

<html>

<head>

<title>HTML span Tag</title>

</head>

<body>

<p>This is <span style="color:red">red</span> and this is <span style="color:green">green</span></p>

</body>

</html>

CHAPTER [15]

# HTML Layouts:

A webpage layout is very important to give better look to your website. It takes considerable time to design a website's layout with great look and feel.

Now a days, all modern websites are using CSS and Javascript based framework to come up with responsive and dynamic websites but you can create a good layout using simple HTML tables or division tags in combination with other formatting tags. This chapter will give you few examples on how to create a simple but working layout for your webpage using pure HTML and its attributes.

# HTML Layout - Using Tables:

The simplest and most popular way of creating layouts is using HTML <table> tag. These tables are arranged in columns and rows, so you can utilize these rows and columns in whatever way you like.

Example:

For example, the following HTML layout example is achieved using a table with 3 rows and 2 columns but the header and footer column spans both columns using the colspan attribute:

<!DOCTYPE html>

<html>

<head>

<title>HTML Layout using Tables</title>

</head>

<body>

<table width="100%" border="0">

<tr>

<td colspan="2" bgcolor="#b5dcb3">

<h1>This is Web Page Main title</h1>

</td>

</tr>

<tr valign="top">

<td bgcolor="#aaa" width="50">

<b>Main Menu</b><br />

HTML<br />

PHP<br />

PERL...

</td>

<td bgcolor="#eee" width="100" height="200">

Technical and Managerial Tutorials

</td>

</tr>

<tr>

<td colspan="2" bgcolor="#b5dcb3">

<center>

Copyright © 2007 Tutorialspoint.com

</center>

</td>

</tr>

</table>

</body>

</html>

# Multiple Columns Layout - Using Tables:

You can design your webpage to put your web content in multiple pages. You can keep your content in middle column and you can use left column to use menu and right column can be used to put advertisement or some other stuff. This layout will be very similar to what we have at our website tutorialspoint.com.

Example:

Here is an example to create three column layout:

<!DOCTYPE html>

<html>

<head>

<title>Three Column HTML Layout</title>

</head>

<body>

<table width="100%" border="0">

<tr valign="top">

<td bgcolor="#aaa" width="20%">

<b>Main Menu</b><br />

HTML<br />

PHP<br />

PERL...

</td>

<td bgcolor="#b5dcb3" height="200" width="60%">

Technical and Managerial Tutorials

</td>

<td bgcolor="#aaa" width="20%">

<b>Right Menu</b><br />

HTML<br />

PHP<br />

PERL...

</td>

</tr>

<table>

</body>

</html>

# HTML Layouts - Using DIV, SPAN:

The <div> element is a block level element used for grouping HTML elements. While the <div> tag is a block-level element, the HTML <span> element is used for grouping elements at an inline level.

Although we can achieve pretty nice layouts with HTML tables, but tables weren't really designed as a layout tool. Tables are more suited to presenting tabular data.

Note:This example makes use of Cascading Style Sheet (CSS), so before understanding this example you need to have a better understanding on how CSS works.

Example:

Here we will try to achieve same result using <div> tag along with CSS, whatever you have achieved using <table> tag in previous example.

<!DOCTYPE html>

<html>

<head>

<title>HTML Layouts using DIV, SPAN</title>

</head>

<body>

<div style="width:100%">

<div style="background-color:#b5dcb3; width:100%">

<h1>This is Web Page Main title</h1>

</div>

<div style="background-color:#aaa; height:200px;width:100px;float:left;">

<div><b>Main Menu</b></div>

HTML<br />

PHP<br />

PERL...

</div>

<div style="background-color:#eee; height:200px;width:350px;float:left;">

<p>Technical and Managerial Tutorials</p>

</div>

<div style="background-color:#aaa; height:200px;width:100px;float:right;">

<div><b>Right Menu</b></div>

HTML<br />

PHP<br />

PERL...

</div>

<div style="background-color:#b5dcb3;clear:both">

<center>

Copyright © 2007 Tutorialspoint.com

</center>

</div>

</div>

</body></html>

CHAPTER [16]

# HTML Iframes:

You can define an inline frame with HTML tag **<iframe>**. The <iframe> tag is not somehow related to <frameset> tag, instead, it can appear anywhere in your document. The <iframe> tag defines a rectangular region within the document in which the browser can display a separate document, including scrollbars and borders.

The **src** attribute is used to specify the URL of the document that occupies the inline frame.

# Iframe Syntax:

The syntax for adding an iframe is:

iframe src="URL"></iframe>

The **src** attribute specifies the URL (web address) of the iframe page.

## **Iframe - Set Height and Width:**

Use the **height** and **width** attributes to specify the size.

The attribute values are specified in pixels by default, but they can also be in percent (like "80%").

Example:

<!DOCTYPE html>

<html>

<body>

<iframe src="demo\_iframe.htm" width="200" height="200"></iframe>

</body>

</html>

# Iframe - Remove the Border:

By default, an iframe has a black border around it.

To remove the border, add the style attribute and use the CSS border property:

Example:

<!DOCTYPE html>

<html>

<body>

<iframe src="demo\_iframe.htm" style="border:none"></iframe>

</body>

</html>

With CSS, you can also change the size, style and color of the iframe's border:

Example:

<!DOCTYPE html>

<html>

<body>

<iframe src="demo\_iframe.htm" style="border:5px dotted red"></iframe>

</body>

</html>

# Use iframe as a Target for a Link:

An iframe can be used as the target frame for a link.

The **target** attribute of the link must refer to the **name** attribute of the iframe:

Example:

<!DOCTYPE html>

<html>

<body>

<iframe width="100%" height="300px" src="demo\_iframe.htm" name="iframe\_a"></iframe>

<p><a href="http:// [www.google.com](http://www.google.com/)" target="iframe\_a">[google.com](http://www.google.com/)</a></p>

<p>When the target of a link matches the name of an iframe, the link will open in the iframe.</p>

</body>

</html>

CHAPTER [17]

# HTML Javascript:

A **script** is a small piece of program that can add interactivity to your website. For example, a script could generate a pop-up alert box message, or provide a dropdown menu. This script could be written using Javascript or VBScript.

You can write various small functions, called event handlers using any of the scripting language and then you can trigger those functions using HTML attributes.

Now a days only **Javascript** and associated frameworks are being used by most of the web developers, VBScript is not even supported by various major browsers.

You can keep Javascript code in a separate file and then include it whereever it's needed, or you can define functionality inside HTML document itself. Let's see both the cases one by one with suitable examples.

# External Javascript:

If you are going to define a functionality which will be used in various HTML documents then it's better to keep that functionality in a separate Javascript file and then include that file in your HTML documents. A Javascript file will have extension as **.js** and it will be included in HTML files using <script> tag.

Example:

Consider we define a small function using Javascript in **script.js** which has following code:

function Hello()

{

alert("Hello, World");

}

Now let's make use of the above external Javascript file in our following HTML document:

<!DOCTYPE html>

<html>

<head>

<title>Javascript External Script</title>

<script src="/html/script.js" type="text/javascript"/></script>

</head>

<body>

<input type="button" onclick="Hello();" name="ok" value="Click Me" />

</body>

</html>

This will produce following result, where you can try to click on the given button:

# Internal Script:

You can write your script code directly into your HTML document. Usually we keep script code in header of the document using <script> tag, otherwise there is no restriction and you can put your source code anywhere in the document but inside <script> tag.

Example:

<!DOCTYPE html>

<html>

<head>

<title>Javascript Internal Script</title>

<base href="http://www.tutorialspoint.com/" />

<script type="text/javascript">

function Hello(){

alert("Hello, World");

}

</script>

</head>

<body>

<input type="button" onclick="Hello();" name="ok" value="Click Me" />

</body>

</html>

# Event Handlers:

Event handlers are nothing but simply defined functions which can be called against any mouse or keyboard event. You can define your business logic inside your event handler which can vary from a single to 1000s of line code.

Following example explains how to write an event handler. Let's write one simple function *EventHandler()* in the header of the document. We will call this function when any user brings mouse over a paragraph.

<!DOCTYPE html>

<html>

<head>

<title>Event Handlers Example</title>

<base href="http://www.google.com/" />

<script type="text/javascript">

function EventHandler(){

alert("I'm event handler!!");

}

</script>

</head>

<body>

<p onmouseover="EventHandler();">Bring your mouse here to see an alert</p>

</body>

</html>

Now this will produce following result. Bring your mouse over this line and see the result:

# Hide Scripts from Older Browsers:

Although most (if not all) browsers these days support Javascript, but still some older browsers don't. If a browser doesn't support JavaScript, instead of running your script, it would display the code to the user. To prevent this, you can simply place HTML comments around the script as shown below.

JavaScript Example:

<script type="text/javascript">

<!--

document.write("Hello Javascript!");

//-->

</script>

VBScript Example:

<script type="text/vbscript">

<!--

document.write("Hello VBScript!")

'-->

</script>

# The <noscript> Element:

You can also provide alternative info to the users whose browsers don't support scripts and for those users who have disabled script option their browsers. You can do this using the **<noscript>** tag.

JavaScript Example:

<script type="text/javascript">

<!--

document.write("Hello Javascript!");

//-->

</script>

<noscript>Your browser does not support Javascript!</noscript>

VBScript Example:

<script type="text/vbscript">

<!--

document.write("Hello VBScript!")

'-->

</script>

<noscript>Your browser does not support VBScript!</noscript>

# Default Scripting Language:

There may be a situation when you will include multiple script files and ultimately using multiple <script> tags. You can specify a default scripting language for all your *script* tags. This saves you from specifying the language everytime you use a script tag within the page. Below is the example:

<meta http-equiv="Content-Script-Type" content="text/JavaScript" />

Note that you can still override the default by specifying a language within the script tag.

# JavaScript can change HTML content:

<!DOCTYPE html>

<html>

<body>

<h1>My First JavaScript</h1>

<p>JavaScript can change the content of an HTML element:</p>

<button type="button" onclick="myFunction()">Click Me!</button>

<p id="demo">This is a demonstration.</p>

<script>

function myFunction() {

document.getElementById("demo").innerHTML = "Hello JavaScript!";

}

</script>

</body>

</html>

# JavaScript can change HTML styles:

<!DOCTYPE html>

<html>

<body>

<h1>My First JavaScript</h1>

<p>JavaScript can change the style of an HTML element:</p>

<button type="button" onclick="myFunction()">Click Me!</button>

<p id="demo">This is a demonstration.</p>

<script>

function myFunction() {

document.getElementById("demo"). style.color = "blue";

}

</script>

</body>

</html>

# JavaScript can change HTML attributes:

<!DOCTYPE html>

<html>

<body>

<script>

function light(sw) {

var pic;

if (sw == 0) {

pic = "pic\_bulboff.gif"

} else {

pic = "pic\_bulbon.gif"

}

document.getElementById('myImage').src = pic;

}

</script>

<img id="myImage" src="pic\_bulboff.gif" width="100" height="180">

<p>

<button type="button" onclick="light(1)">Light On</button>

<button type="button" onclick="light(0)">Light Off</button>

</p>

</body>

</html>

CHAPTER [17]

# HTML Head:

# The HTML <head> Element:

The <head> element is a container for metadata (data about data).

HTML metadata is data about the HTML document. Metadata is not displayed.

Metadata typically define document title, styles, links, scripts, and other meta information.

The following tags describe metadata: <title>, <style>, <meta>, <link>, <script>, and <base>.

# Omitting <html> and <body>?

In the HTML5 standard, the <html> tag, the <body> tag, and the <head> tag can be omitted.

The following code will validate as HTML5:

The <html> element is the document root. It is the recommended place for specifying the page language:

<!DOCTYPE html>  
<html lang="en-US">

Declaring a language is important for accessibility applications (screen readers) and search engines.

Omitting <html> and <body> can crash badly-written DOM/XML software.

Finally, omitting <body> can produce errors in older browsers (IE9).

# Omitting <head>:

In the HTML5 standard, the <head> tag can also be omitted.

By default, browsers will add all elements before <body>, to a default <head> element.

You can reduce the complexity of HTML, by omitting the <head> tag:

Example:

<!DOCTYPE html>

<html>

<title>Page Title</title>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

# The HTML <title> Element:

The <title> element defines the title of the document.

The <title> element is required in all HTML/XHTML documents.

The <title> element:

* defines a title in the browser tab
* provides a title for the page when it is added to favorites
* displays a title for the page in search engine results

A simplified HTML document:

<!DOCTYPE html>

<html>

<title>Page Title</title>

<body>

<p>The content of the body element is displayed in the browser window.</p>

<p>The content of the title element is displayed in the browser tab, in favorites and in search engine results.</p>

</body>

</html>

# The HTML <style> Element:

The <style> element is used to define style information for an HTML document.

Inside the <style> element you specify how HTML elements should render in a browser:

<!DOCTYPE html>

<html>

<title>Page Title</title>

<style>

body {background-color:yellow;}

p {color:blue;}

</style>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

# The HTML <link> Element:

The <link> element defines the page relationship to an external resource.

The <link> element is most often used to link to style sheets:

<!DOCTYPE html>

<html>

<title>Page Title</title>

<link rel="stylesheet" href="mystyle.css">

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

# The HTML <meta> Element:

The <meta> element is used to specify page description, keywords, author, and other metadata.

Metadata is used by browsers (how to display content), by search engines (keywords), and other web services.

Define keywords for search engines:

<meta name="keywords" content="HTML, CSS, XML, XHTML, JavaScript">

Define a description of your web page:

<meta name="description" content="Free Web tutorials on HTML and CSS">

Define the character set used:

<meta charset="UTF-8">

Define the author of a page:

<meta name="author" content="Hege Refsnes">

Refresh document every 30 seconds:

<meta http-equiv="refresh" content="30">

CHAPTER [18]

HTML Forms:

HTML Forms are required when you want to collect some data from the site visitor. For example during user registration you would like to collect information such as name, email address, credit card, etc.

A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc. The back-end application will perform required processing on the passed data based on defined business logic inside the application.

There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML <form> tag is used to create an HTML form and it has following syntax:

Syntax:

<form action="Script URL" method="GET|POST">

form elements like input, textarea etc.

</form>

# The <input> Element

The **<input>** element is the most important **form element**.

The <input> element has many variations, depending on the **type** attribute.

# Text Input

**<input type="text">** defines a one-line input field for **text input**:

Example:

<form>  
  First name:<br>  
  <input type="text" name="firstname">  
  <br>  
  Last name:<br>  
  <input type="text" name="lastname">  
</form>

# Radio Button Input

**<input type="radio">** defines a **radio button**.

Radio buttons let a user select ONE of a limited number of choices:

<form>  
  <input type="radio" name="sex" value="male" checked>Male  
  <br>  
  <input type="radio" name="sex" value="female">Female  
</form>

# The Submit Button

**<input type="submit">** defines a button for **submitting** a form to a **form-handler**.

The form-handler is typically a server page with a script for processing input data.

The form-handler is specified in the form's **action** attribute:

Example:

<form action="action\_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey">  
  <br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse">  
  <br><br>  
  <input type="submit" value="Submit">  
</form>

# The Action Attribute

The **action attribute** defines the action to be performed when the form is submitted.

The common way to submit a form to a server, is by using a submit button.

Normally, the form is submitted to a web page on a web server.

In the example above, a server-side script is specified to handle the submitted form:

<form **action=**"action\_page.php">

If the action attribute is omitted, the action is set to the current page.

# The Method Attribute

The **method attribute** specifies the HTTP method (**GET** or **POST**) to be used when submitting the forms:

<form action="action\_page.php" method="get">

Or

<form action="action\_page.php" **method=**"post">

# When to Use GET?

You can use GET (the default method):

If the form submission is passive (like a search engine query), and without sensitive information.

When you use GET, the form data will be visible in the page address:

action\_page.php?firstname=Mickey&lastname=Mouse

# When to Use POST?

You should use POST:

If the form is updating data, or includes sensitive information (password).

POST offers better security because the submitted data is not visible in the page address.

# The Name Attribute

To be submitted correctly, each input field must have a name attribute.

This example will only submit the "Last name" input field:

Example:

<form action="action\_page.php">  
  First name:<br>  
  <input type="text" value="Mickey">  
  <br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse">  
  <br><br>  
  <input type="submit" value="Submit">  
</form>

# Grouping Form Data with <fieldset>

The **<fieldset>** element groups related data in a form.

The **<legend>** element defines a caption for the <fieldset> element.

Example:

<form action="action\_page.php">  
  <fieldset>  
    <legend>Personal information:</legend>  
    First name:<br>  
    <input type="text" name="firstname" value="Mickey">  
    <br>  
    Last name:<br>  
    <input type="text" name="lastname" value="Mouse">  
    <br><br>  
    <input type="submit" value="Submit">  
  </fieldset>  
</form>

# HTML Form Attributes

An HTML <form> element, with all possible attributes set, will look like this:

<form action="action\_page.php" method="post" target="\_blank" accept-charset="UTF-8"  
enctype="application/x-www-form-urlencoded" autocomplete="off" novalidate>  
.  
form elements  
.  
</form>

HTML Form Elements:

# The <input> Element

The most important form element is the **<input>** element.

The <input> element can vary in many ways, depending on the **type** attribute.

# The <select> Element (Drop-Down List)

The **<select>** element defines a **drop-down** list:

Example:

<select name="cars">  
  <option value="volvo">Volvo</option>  
  <option value="saab">Saab</option>  
  <option value="fiat">Fiat</option>  
  <option value="audi">Audi</option>  
</select>

The **<option>** elements defines the options to select.

The list will normally show the first item as selected.

You can add a selected attribute to define a predefined option.

Example:

<option value="fiat" selected>Fiat</option>

# The <textarea> Element

The **<textarea>** element defines a multi-line input field (**a text area**):

Example:

<textarea name="message" rows="10" cols="30">  
The cat was playing in the garden.  
</textarea>

# The <button> Element

The **<button>** element defines a clickable **button**:

Example:

<button type="button" onclick="alert('Hello World!')">Click Me!</button>

# HTML5 Form Elements

HTML5 added the following form elements:

* <datalist>
* <output>

# HTML5 <datalist> Element

The **<datalist>** element specifies a list of pre-defined options for an <input> element.

Users will see a drop-down list of pre-defined options as they input data.

The **list** attribute of the <input> element, must refer to the **id** attribute of the <datalist> element.

Example:

An <input> element with pre-defined values in a <datalist>:

<form action="action\_page.php">  
  <input list="browsers">  
  <datalist id="browsers">  
    <option value="Internet Explorer">  
    <option value="Firefox">  
    <option value="Chrome">  
    <option value="Opera">  
    <option value="Safari">  
  </datalist>   
</form>

# HTML5 <output> Element

The <output> element represents the result of a calculation (like one performed by a script).

Example:

Perform a calculation and show the result in an <output> element:

<form action="action\_page.asp"  
  oninput="x.value=parseInt(a.value)+parseInt(b.value)">  
  0  
  <input type="range"  id="a" name="a" value="50">  
  100 +  
  <input type="number" id="b" name="b" value="50">  
  =  
  <output name="x" for="a b"></output>  
  <br><br>  
  <input type="submit">  
</form>

# Input Type: text

**<input type="text">** defines a one-line input field for **text input**:

Example:

<form>  
First name:<br>  
<input type="text" name="firstname">  
<br>  
Last name:<br>  
<input type="text" name="lastname">  
</form>

# Input Type: password

**<input type="password">** defines a **password field**:

Example:

|  |  |
| --- | --- |
| <form> User name:<br> <input type="text" name="username"> <br> User password:<br> <input type="password" name="psw"> </form> |  |

# Input Type: submit

**<input type="submit">** defines a button for **submitting** form input to a **form-handler**.

The form-handler is typically a server page with a script for processing input data.

If you omit the submit button's value attribute, the button will get a default text:

Example:

<form action="action\_page.php">  
First name:<br>  
<input type="text" name="firstname" value="Mickey">  
<br>  
Last name:<br>  
<input type="text" name="lastname" value="Mouse">  
<br><br>  
<input type="submit">  
</form>

# Input Type: radio

**<input type="radio">** defines a **radio button**.

Radio buttons let a user select ONLY ONE of a limited number of choices:

Example:

<form>  
<input type="radio" name="sex" value="male" checked> Male  
<br>  
<input type="radio" name="sex" value="female"> Female  
</form>

# Input Type: checkbox

**<input type="checkbox">** defines a **checkbox**.

Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example:

<form>  
<input type="checkbox" name="vehicle1" value="Bike"> I have a bike  
<br>  
<input type="checkbox" name="vehicle2" value="Car"> I have a car   
</form>

# Input Type: button

**<input type="button">** defines a **button**:

Example:

<input type="button" onclick="alert('Hello World!')" value="Click Me!">

# HTML5 Input Types

HTML5 added several new input types:

* color
* date
* datetime
* datetime-local
* email
* month
* number
* range
* search
* tel
* time
* url
* week

# Input Type: number

The **<input type="number">** is used for input fields that should contain a numeric value.

You can set restrictions on the numbers.

Example:

<form>  
  Quantity (between 1 and 5):  
  <input type="number" name="quantity" min="1" max="5">  
</form>

Example:

<form>  
  Quantity:  
  <input type="number" name="points" min="0" max="100" step="10" value="30">  
</form>

# Input Type: date

The **<input type="date">** is used for input fields that should contain a date.

Depending on browser support, a date picker can show up in the input field.

Example:

<form>  
  Birthday:  
  <input type="date" name="bday">  
</form>

Example:

<form>  
  Enter a date before 1980-01-01:  
  <input type="date" name="bday" max="1979-12-31"><br>  
  Enter a date after 2000-01-01:  
  <input type="date" name="bday" min="2000-01-02"><br>  
</form>

# Input Type: color

The **<input type="color">** is used for input fields that should contain a color.

Depending on browser support, a color picker can show up in the input field.

Example:

<form>  
  Select your favorite color:  
  <input type="color" name="favcolor">  
</form>

# Input Type: range

The **<input type="range">** is used for input fields that should contain a value within a range.

Depending on browser support, the input field can be displayed as a slider control.

Example:

<form>  
  <input type="range" name="points" min="0" max="10">  
</form>

You can use the following attributes to specify restrictions: min, max, step, value.

# Input Type: month

The **<input type="month">** allows the user to select a month and year.

Depending on browser support, a date picker can show up in the input field.

Example:

<form>  
  Birthday (month and year):  
  <input type="month" name="bdaymonth">  
</form>

# Input Type: week

The **<input type="week">** allows the user to select a week and year.

Depending on browser support, a date picker can show up in the input field.

Example:

<form>  
  Select a week:  
  <input type="week" name="week\_year">  
</form>

# Input Type: time

The **<input type="time">** allows the user to select a time (no time zone).

Depending on browser support, a time picker can show up in the input field.

Example:

<form>  
  Select a time:  
  <input type="time" name="usr\_time">  
</form>

# Input Type: datetime

The **<input type="datetime">** allows the user to select a date and time (with time zone).

Example:

<form>  
  Birthday (date and time):  
  <input type="datetime" name="bdaytime">  
</form>

# Input Type: datetime-local

The **<input type="datetime-local">** allows the user to select a date and time (no time zone).

Depending on browser support, a date picker can show up in the input field.

Example:

<form>  
  Birthday (date and time):  
  <input type="datetime-local" name="bdaytime">  
</form>

# Input Type: email

The **<input type="email">** is used for input fields that should contain an e-mail address.

Depending on browser support, the e-mail address can be automatically validated when submitted.

Some smartphones recognize the email type, and adds ".com" to the keyboard to match email input.

Example:

<form>  
  E-mail:  
  <input type="email" name="email">  
</form>

# Input Type: search

The **<input type="search">** is used for search fields (a search field behaves like a regular text field).

Example:

<form>  
  Search Google:  
  <input type="search" name="googlesearch">  
</form>

# Input Type: tel

The **<input type="tel">** is used for input fields that should contain a telephone number.

The tel type is currently supported only in Safari 8.

Example:

<form>  
  Telephone:  
  <input type="tel" name="usrtel">  
</form>

# Input Type: url

The **<input type="url">** is used for input fields that should contain a URL address.

Depending on browser support, the url field can be automatically validated when submitted.

Some smartphones recognize the url type, and adds ".com" to the keyboard to match url input.

Example:

<form>  
  Add your homepage:  
  <input type="url" name="homepage">  
</form>

HTML Input Attributes:

# The value Attribute

The **value** attribute specifies the initial value for an input field:

Example:

<form action="">  
First name:<br>  
<input type="text" name="firstname" value="John">  
<br>  
Last name:<br>  
<input type="text" name="lastname">  
</form>

# The readonly Attribute

The **readonly** attribute specifies that the input field is read only (cannot be changed):

Example:

<form action="">  
First name:<br>  
<input type="text" name="firstname" value="John" readonly>  
<br>  
Last name:<br>  
<input type="text" name="lastname">  
</form>

# The disabled Attribute

The **disabled** attribute specifies that the input field is disabled.

A disabled element is un-usable and un-clickable.

Disabled elements will not be submitted.

Example:

<form action="">  
First name:<br>  
<input type="text" name="firstname" value="John" disabled>  
<br>  
Last name:<br>  
<input type="text" name="lastname">  
</form>

# The size Attribute

The **size** attribute specifies the size (in characters) for the input field:

Example:

<form action="">  
First name:<br>  
<input type="text" name="firstname" value="John" size="40">  
<br>  
Last name:<br>  
<input type="text" name="lastname">  
</form>

# The maxlength Attribute

The **maxlength** attribute specifies the maximum allowed length for the input field:

Example:

<form action="">  
First name:<br>  
<input type="text" name="firstname" maxlength="10">  
<br>  
Last name:<br>  
<input type="text" name="lastname">  
</form>

With a maxlength attribute, the input control will not accept more than the allowed number of characters.

The attribute does not provide any feedback. If you want to alert the user, you must write JavaScript code.

# The autocomplete Attribute

The autocomplete attribute specifies whether a form or input field should have autocomplete on or off.

When autocomplete is on, the browser automatically complete values based on values that the user has entered before.

Note: It is possible to have autocomplete "on" for the form, and "off" for specific input fields, or vice versa.

The autocomplete attribute works with <form> and the following <input> types: text, search, url, tel, email, password, datepickers, range, and color.

Example:

An HTML form with autocomplete on (and off for one input field):

<form action="action\_page.php" autocomplete="on">  
  First name:<input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  E-mail: <input type="email" name="email" autocomplete="off"><br>  
  <input type="submit">  
</form>

# The novalidate Attribute

The novalidate attribute is a <form> attribute.

When present, novalidate specifies that form data should not be validated when submitted.

Example:

Indicates that the form is not to be validated on submit:

<form action="action\_page.php" novalidate>  
  E-mail: <input type="email" name="user\_email">  
  <input type="submit">  
</form>

# The autofocus Attribute

The autofocus attribute is a boolean attribute.

When present, it specifies that an <input> element should automatically get focus when the page loads.

Example:

Let the "First name" input field automatically get focus when the page loads:

First name:

<input type="text" name="fname" autofocus>

# The form Attribute

The form attribute specifies one or more forms an <input> element belongs to.

Note: To refer to more than one form, use a space-separated list of form ids.

Example:

An input field located outside the HTML form (but still a part of the form):

<form action="action\_page.php" id="form1">  
  First name: <input type="text" name="fname"><br>  
  <input type="submit" value="Submit">  
</form>  
  
Last name: <input type="text" name="lname" form="form1">

# The formaction Attribute

The formaction attribute specifies the URL of a file that will process the input control when the form is submitted.

The formaction attribute overrides the action attribute of the <form> element.

The formaction attribute is used with type="submit" and type="image".

Example:

An HTML form with two submit buttons, with different actions:

<form action="action\_page.php">  
  First name: <input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  <input type="submit" value="Submit"><br>  
  <input type="submit" formaction="demo\_admin.asp"  
  value="Submit as admin">  
</form>

# The formenctype Attribute

The formenctype attribute specifies how the form-data should be encoded when submitting it to the server (only for forms with method="post").

The formenctype attribute overrides the enctype attribute of the <form> element.

The formenctype attribute is used with type="submit" and type="image".

Example:

Send form-data that is default encoded (the first submit button), and encoded as "multipart/form-data" (the second submit button):

<form action="demo\_post\_enctype.asp" method="post">  
  First name: <input type="text" name="fname"><br>  
  <input type="submit" value="Submit">  
  <input type="submit" formenctype="multipart/form-data"  
  value="Submit as Multipart/form-data">  
</form>

# The formmethod Attribute

The formmethod attribute defines the HTTP method for sending form-data to the action URL.

The formmethod attribute overrides the method attribute of the <form> element.

The formmethod attribute can be used with type="submit" and type="image".

Example:

The second submit button overrides the HTTP method of the form:

<form action="action\_page.php" method="get">  
  First name: <input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  <input type="submit" value="Submit">  
  <input type="submit" formmethod="post" formaction="demo\_post.asp"  
  value="Submit using POST">  
</form>

# The formnovalidate Attribute

The novalidate attribute is a boolean attribute.

When present, it specifies that the <input> element should not be validated when submitted.

The formnovalidate attribute overrides the novalidate attribute of the <form> element.

The formnovalidate attribute can be used with type="submit".

Example:

A form with two submit buttons (with and without validation):

<form action="action\_page.php">  
  E-mail: <input type="email" name="userid"><br>  
  <input type="submit" value="Submit"><br>  
  <input type="submit" formnovalidate value="Submit without validation">  
</form>

# The formtarget Attribute

The formtarget attribute specifies a name or a keyword that indicates where to display the response that is received after submitting the form.

The formtarget attribute overrides the target attribute of the <form> element.

The formtarget attribute can be used with type="submit" and type="image".

Example:

A form with two submit buttons, with different target windows:

<form action="action\_page.php">  
  First name: <input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  <input type="submit" value="Submit as normal">  
  <input type="submit" formtarget="\_blank"  
  value="Submit to a new window">  
</form>

# The height and width Attributes

The height and width attributes specify the height and width of an <input> element.

The height and width attributes are only used with <input type="image">.

Example:

Define an image as the submit button, with height and width attributes:

<input type="image" src="img\_submit.gif" alt="Submit" width="48" height="48">

# The list Attribute

The list attribute refers to a <datalist> element that contains pre-defined options for an <input> element.

Example:

An <input> element with pre-defined values in a <datalist>:

<input list="browsers">  
<datalist id="browsers">  
  <option value="Internet Explorer">  
  <option value="Firefox">  
  <option value="Chrome">  
  <option value="Opera">  
  <option value="Safari">  
</datalist>

# The min and max Attributes

The min and max attributes specify the minimum and maximum value for an <input> element.

The min and max attributes work with the following input types: number, range, date, datetime, datetime-local, month, time and week.

Example:

<input> elements with min and max values:

Enter a date before 1980-01-01:  
<input type="date" name="bday" max="1979-12-31">  
  
Enter a date after 2000-01-01:  
<input type="date" name="bday" min="2000-01-02">  
  
Quantity (between 1 and 5):  
<input type="number" name="quantity" min="1" max="5">

# The multiple Attribute

The multiple attribute is a boolean attribute.

When present, it specifies that the user is allowed to enter more than one value in the <input> element.

The multiple attribute works with the following input types: email, and file.

Example:

A file upload field that accepts multiple values:

Select images: <input type="file" name="img" multiple>

# The pattern Attribute

The pattern attribute specifies a regular expression that the <input> element's value is checked against.

The pattern attribute works with the following input types: text, search, url, tel, email, and password.

Example:

An input field that can contain only three letters (no numbers or special characters):

Country code:

<input type="text" name="country\_code" pattern="[A-Za-z]{3}" title="Three letter country code">

# The placeholder Attribute

The placeholder attribute specifies a hint that describes the expected value of an input field (a sample value or a short description of the format).

The hint is displayed in the input field before the user enters a value.

The placeholder attribute works with the following input types: text, search, url, tel, email, and password.

Example:

<input type="text" name="fname" placeholder="First name">

# The required Attribute

The required attribute is a boolean attribute.

When present, it specifies that an input field must be filled out before submitting the form.

The required attribute works with the following input types: text, search, url, tel, email, password, date pickers, number, checkbox, radio, and file.

Example:

Username: <input type="text" name="usrname" required>

# The step Attribute

The step attribute specifies the legal number intervals for an <input> element.

Example: if step="3", legal numbers could be -3, 0, 3, 6, etc.

Note: The step attribute can be used together with the max and min attributes to create a range of legal values.

The step attribute works with the following input types: number, range, date, datetime, datetime-local, month, time and week.

Example:

An input field with a specified legal number intervals:

<input type="number" name="points" step="3">